INTERFACE AGE™

COMPUTING FOR HOME AND BUSINESS APPLICATIONS

VOLUME 4, ISSUE 6 JUNE 1979 $2.00
CANADA/MEXICO $2.50 INTERNATIONAL $4.50

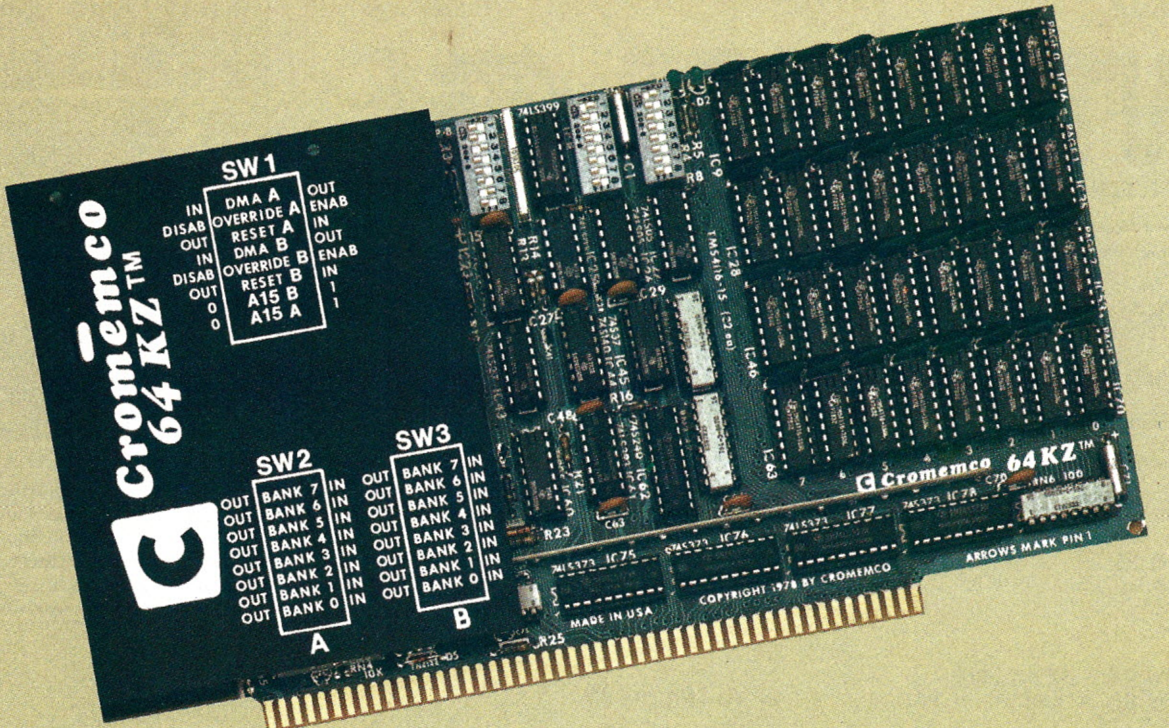
DESK COPY

THE AUTOMATED HOME

VOICE SYNTHESIS

PASCAL NOTEBOOK

NTS mini series
UNIT #4



You can do surprising things when you have 64 kilobytes of fast RAM on one card

4 MHz FAST—AND EXPANDABLE

Here's 64 kilobytes of memory on one RAM card. Yes, we mean 512K bits of read/write memory on this single card.

And, yes, we mean it's fast. With 150-nanosecond chip access times — so the card can operate in fast Z-80 systems with no wait states. Repeat, no wait states.

EXPANDABLE ON TWO LEVELS

Not only does the new Model 64KZ give you a large, fast RAM but it is expandable on two levels.

First, through our Cromemco Bank Select feature, you can expand to 512 kilobytes in eight 64K banks.

Or, with our Extended Bank Select feature, you can expand memory space to as much as 16 megabytes.

This expandability we call your obsolescence insurance.

The legend on the card's heat sink is an easy reference for address and bank selection.

BENCHMARK IT

Obviously, the speed and memory capacity of this new card give you a lot of power.

You can see that for yourself in our new 7-station Multi-User Computer System which uses these Model 64KZ cards. This S100-bus system outperforms the speed of many if not most timesharing systems of up to 10 times the Cromemco price.

And yet where some of these much more expensive and cumbersome systems clearly slow to a snail's pace when timesharing, the Cromemco system using Bank Select switching runs surprisingly fast.

SEE IT NOW

See the new Model 64KZ at your computer dealer now. Study the literature on it. See how for only $1785 you can get around that ever-present barrier of memory that's too little and too slow.

For high reliability all Cromemco memory cards are burned in at the factory in these temperature-controlled ovens.

Cromemco Multi-User System shown with 7 stations

Cromemco
incorporated
280 BERNARDO AVE., MOUNTAIN VIEW, CA 94040 • (415) 964-7400
Tomorrow's computers now

CIRCLE INQUIRY NO. 16

VOL. 4, ISSUE 6 JUNE 1979
SINCE DECEMBER 1975

INTERFACE AGE™

COMPUTING FOR HOME AND BUSINESS APPLICATIONS

GENERAL FEATURES

BUSINESS FEATURES

HARDWARE FEATURES

TUTORIALS

SOFTWARE FEATURES

FREE RUNNING

DEPARTMENTS

THIS MONTH'S COVER

Designed by Fino Ortiz, Art Director. Photography by Don May. House provided by Alice's Imagination Shop of Torrance, CA. Terminal supplied by Vector Graphic Inc. of Westlake Village, CA.

PUBLISHERS
PUBLISHER **ROBERT S. JONES**
EXECUTIVE PUBLISHER **NANCY A. JONES**

ADMINISTRATION
PUBLICATION DIRECTOR **MIKE ANTICH**
PUBLICATION ASSISTANT **DORIS RIOPEL**
ACCOUNTING SUPERVISOR **KAY SOTO**
ACCOUNTING ASSISTANTS **MARY ANN LOWER, SHIRLEY MAZENKO**
ASSISTANT-TO-THE-PUBLISHER **JOANNA KONDRATH**
CIRCULATION SECRETARY **JETTA ANDRIONI**
CIRCULATION ASSISTANTS **CHARLOTTE SEVEDGE, DOTTIE SMITH**

EDITORIAL
EDITOR-IN-CHIEF **CARL WARREN**
SENIOR EDITOR SOUTHEASTERN REGION **BILL TURNER**
ASSISTANT EDITOR **TERRY COSTLOW**
ASSOCIATE EDTIORS **JIM SCHREIER, HENRY DAVIS**
NORTHWESTERN REGIONAL EDITOR **ADAM OSBORNE, PhD**
NORTHEASTERN REGIONAL EDITOR **ROGER C. GARRETT**
SYSTEMS EDITOR **TOM FOX**
SOFTWARE EDITOR **ALAN R. MILLER, PhD**
CONTRIBUTING EDITORS **R. H. DISTLER, R. W. BEMER**
STAFF REPORTER **BETSY GILBERT**
Editorial Correspondence
Direct all correspondence to the appropriate editor at: INTERFACE AGE Magazine, P.O. Box 1234, Cerritos, CA 90701.

PRODUCTION
PRODUCTION MANAGER **DAVE ANDERSON**
ASSISTANT PRODUCTION MANAGER **SHELLEY WRIGHT**
ART DIRECTOR **FINO ORTIZ**
ARTIST **SAMANTHA LEE**
TYPOGRAPHER **MELODY A. MARTENS**

ADVERTISING
NEW ENGLAND REGION **DICK GREEN**
7 Lincoln St., Wakefield, MA 01880 (617) 245-9105
EASTERN REGION **PAUL CONFREY**
20 Community Pl., Morristown, NJ 07960 (201) 267-3032
MIDWEST REGION **AL GRAVENHORST & STEVE SKINNER**
5901 N. Cicero Ave., Chicago, IL 60646 (312) 545-8621
WESTERN REGION **BRUCE BERKEY & ZACH BOVINETTE**
61 S. Lake Ave., Pasadena, CA 91106 (213) 795-7002
COMPUTER RETAIL STORES NATIONWIDE
CALL (213) 795-7002 (COLLECT)

INTERFACE AGE EUROPE
DIRECTOR, EUROPEAN OPERATIONS **H.L. GROHMANN**
Birkenweg 43, 8051 Unterzolling, West Germany
Telephone: 8167-366

FOREIGN CIRCULATION
JAPAN CIRCULATION **KAZUHIKO NISHI**
ASCII Publishing, 305 HI TORIO, 5-6-4 Minami Aoyama, Minato-ku, Tokyo 107 Japan
Telephone: (03) 407-4910
UNITED KINGDOM CIRCULATION **VINCENT COEN**
L.P. Enterprise, 313 Kingston Road, Ilford, Essex, England IGIIPJ
Telephone: 01-553-1001
FRANCE CIRCULATION **ROLAND HESSE**
Euro Computer Shop Paris, 16, Rue Louis Pasteur, 92100 Boulogne, France
Telephone: Paris 825-82-52
WESTERN CANADA CIRCULATION **BRIAN I.J. WIEBE**
Kitronic, 25236 26th Avenue RR5, Aldergrove, B.C. VOX 1A0
Telephone: (604) 856-2301
EASTERN CANADA CIRCULATION **LIZ JANEK**
RS-232, Ltd., 186 Queen Street West, Toronto, Ontario M5V 1Z1
Telephone: (416) 598-0288
AUSTRALIA CIRCULATION **R. J. HOESS**
Electronic Concepts Pty. Ltd., 52-58 Clarence Street, Sydney NSW 2000
Telephone: 29-3753
INDONESIA, SINGAPORE, MALAYSIA CIRCULATION **LEE MILES**
The Computer Centre, Pte. Ltd., 5366, Woh Hup Complex, Beach Rd. Singapore 7
Telephone: 293-2630

MEMBER OF THE WESTERN PUBLICATIONS ASSOCIATION

AMERICAN SOCIETY OF BUSINESS PRESS EDITORS

INTERFACE AGE Magazine, published monthly by McPheters, Wolfe & Jones, 16704 Marquardt Ave., Cerritos, CA 90701. Subscription rates: U.S. $18.00, Canada/Mexico $20.00, all other countries $28.00. Make checks payable in U.S. funds drawn on a U.S. bank. Opinions expressed in by-lined articles do not necessarily reflect the opinion of this magazine or the publisher. Mention of products by trade name in editorial material or advertisements contained herein in no way constitutes endorsement of the product or products by this magazine or the publisher.

INTERFACE AGE Magazine is catalogued in the Library of Congress, Classification No. QA75.5.155. ISSN Publication No. 0147-2992. Membership in Audit Bureau of Circulations applied for.
POSTMASTER: Please send change of address form 3579 and undelivered copies to **INTERFACE AGE** Magazine, 16704 Marquardt Ave., Cerritos, CA 90701. Second-class postage paid at Artesia, California 90701 and at additional mailing offices.

CAST A SPELL.
WIN A SORCERER.

A FANTASTIC SHOW

This past March 22-25, The Gutenberg Festival was held at the Long Beach Convention Center here in California. This festival is for the graphics arts industry and primarily features all the gadgets that printers, artists and publishers use in their businesses. What made this show unusual was the number of computer companies that were exhibiting.

The main thrust of the computer exhibits were word processing and typesetting machines. However, general business machines were very much in evidence. One such exhibit was R.B. Graphic Supply, 7291 Garden Grove Boulevard, #1, Garden Grove, CA 92641, (714) 749-9451, Attn: Richard Brock, president. R.B. mainly supplies material to printers and magazines, but has recently begun marketing an Alpha-Micro system with the graphics arts industry in mind. They currently are offering business packages and an estimating package to assist small printing companies in figuring prices.

Along the same lines, the show manager David Jacobson commissioned a program for his Alpha-Micro to handle the show registration plus provide up-to-the-minute management reports. The software, created by Tom Fox of Fox and Associates, is proprietary to the *Horsetrader Magazine,* but from what I understand may be available to other show promoters for a price. The system is time sharing and uses four input terminals and four printers to print the different classification of badges.

Those of you who might be interested in this program or exhibiting next year at the festival should contact: David Jacobson, *Horsetrader Magazine,* P.O. Box 11712, Santa Ana, CA 92711 or call (714) 832-0628. If you're looking for new markets, and want to be a part of possibly the best business show around, you will want to get in touch with Dave as soon as possible, since this show sells out very quickly.

TRAVELING AROUND

During the last several weeks, I have had the chance to travel to Silicon Gulch and visit a number of companies. This is something that I will be doing on a regular basis for the rest of the year so we can keep on top of what's happening.

This last trip took me first to Godbout Electronics in the Oakland Airport.

Bill Godbout's company is housed in a barrack-like building with no apparent signs of who or what is contained inside. Once in the doors, you are immediately struck by the lack of pretentious overhead. As Bill told me on the phone, "When you come up, you will see a large messy surplus operation." He wasn't kidding.

One of the most important things that meets you at Godbout Electronics is the friendly atmosphere, and as you can see from Bill's photo, he presents a jovial, likable appearance.

Bill spends 145% of his time working with suppliers and customers to coordinate his fast paced yet unstructured business. Bill has taken great pains to ensure proper lead times on parts, and to avoid price gouging which is so prevalent in the industry today.

On the tour of the plant, it became quickly apparent that product quality and customer satisfaction were of prime importance. Each order is checked and double checked to make sure that it is filled properly, and potential problems are corrected before the product leaves the shop.

PHOTO 1

One way that Bill and his group prevent problems is by doing a 100% quality control check on all boards that are manufactured. This quality control takes place three times: at the fabricating plants in Arkansas and Reno, and once again at the main office

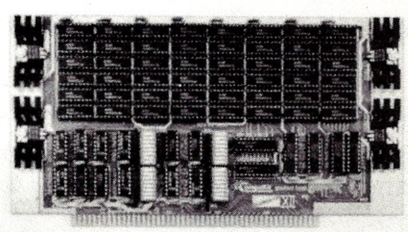


PHOTO 2

in Oakland. All rejects are sent to Bill so that corrective measures may be taken to smooth the production process. Godbout enjoys less than one reject in one hundred boards for each product produced.

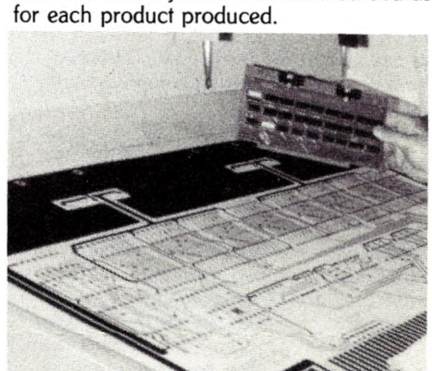


PHOTO 3

 CIRCLE INQUIRY NO. 58

One of the boards that Godbout supplies is the Compukit bank select memory board, shown in Photo 2. This board is available in 16K and 24K, ranging from $369 to $539. The design of the board, as with all the Godbout boards, was not an overnight affair, as can be seen in Photo 3.

Each board is thoroughly engineered to meet both bus and electrical standards, plus all design criteria are completely documented from the R&D stage up to final production.

Godbout Electronics is different than most of the companies currently serving the microcomputer industry in that it is over ten years old. Bill, who is primarily a circuit designer who spent some time with IBM, opened a retail mail order operation in 1973, supplying surplus parts and supplying companies like Poly Paks. His first introduction into the micro world was when he obtained some 8008s and was able to supply them to the first real hobbyists.

Although the computer side of the business accounts for only approximately 20% of the total Godbout picture, Bill believes in supporting the industry. He does this by supporting magazines, clubs, and making sure that the parts he supplies are of top grade.

So if you're in the neighborhood, or want to find out more about Bill's products, you can contact him at Godbout Electronics, Box 2355, Oakland Airport, CA 94614, phone (415) 938-6295.

Across the bay is Cromemco, Inc. at 280 Bernardo Avenue, Mountain View, CA 94043, phone (415) 964-7400. Started just four years ago by Harry Garland and Roger Mellon, both electronic engineers and past members of the Stanford faculty, the company now sports a new building and a 150-employee staff of engineers and fabricators.

PHOTO 4

According to Garland, president of the company, Cromemco provides quality micros for the industrial and scientific markets. When I asked him about his price structure he said: "Our prices are higher because we put a great deal of emphasis on proper engineering and quality bill of materials."

Even though Garland's statement sounds like what a company president would say, it happens to be true. More hours are spent in the design and implementation of the design than can be imagined. Because both Garland and Mellon are enginers of the highest caliber, the whole functional work flow and testing follows a logical path. Each function of a board is carefully tested and burnt in before being incorporated into a system. The system is then run through rigorous testing as a unit before shipment is made.

The software supplied by Cromemco comes from Microsoft, specifically the COBOL and FORTRAN. The multi-user BASIC was developed in-house and supports up to seven users. They are currently developing a Pascal package to be released in the next several months.

All documentation is prepared under the direction of Dr. Alice Ahlgren, whose forte is communication. Dr. Ahlgren's attention to detail and ease of use of the manuals clearly shows in all the documentation supplied by Cromemco.

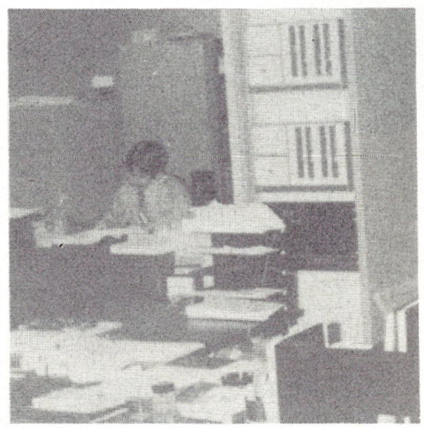


PHOTO 5

One significant if not exciting aspect of Cromemco is that they use their machines

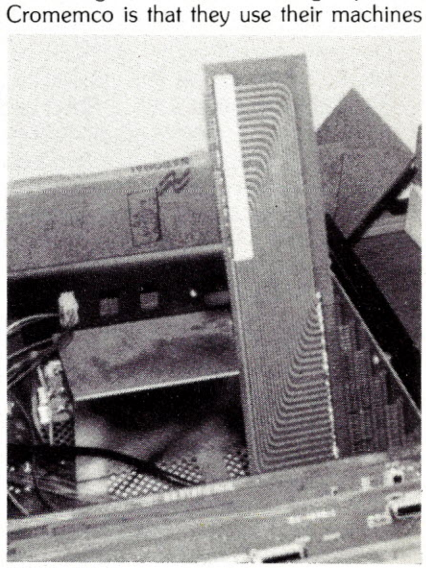
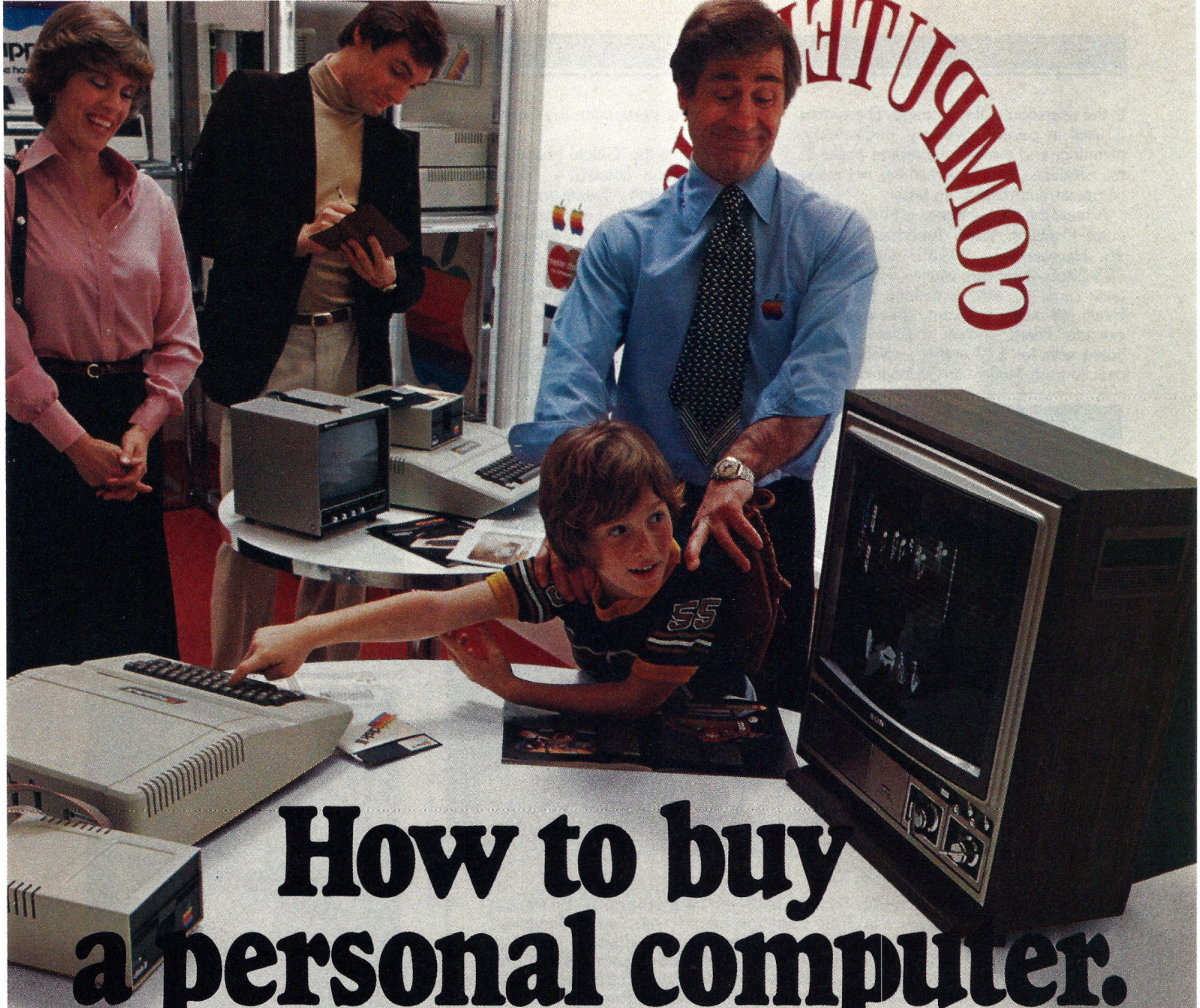


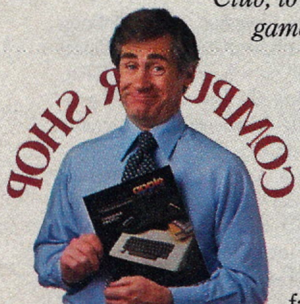
PHOTO 6



How to buy a personal computer.

In California, a store owner charts sales on his Apple Computer. On weekends though, he totes Apple home to help plan family finances with his wife. And for the kids to explore the new world of personal computers.

A hobbyist in Michigan starts a local Apple Computer Club, to challenge other members to computer games of skill and to trade programs.



Innovative folks everywhere have discovered that the era of the personal computer has already begun—with Apple.

Educators and students use Apple in the classroom. Businessmen trust Apple with the books. Parents are making Apple the newest family pastime. And kids of all ages are finding how much fun computers can be, and have no time for TV once they've discovered Apple.

Visit your local computer store

The excitement starts in your local computer store. It's a friendly place, owned by one of your neighbors. He'll show you exactly what you can use a personal computer for.

What to look for

Your local computer store has several different brands to show you. So the salesman can recommend the one that best meets your needs. Chances are, it will be an Apple Computer. Apple is the one you can program yourself. So there's no limit to the things you can do. Most important, Apple's the one with more expansion capability. That means a lot. Because the more you use your Apple, the more uses you'll discover. So your best bet is a personal computer that can grow with you as your skill and involvement grow. Apple's the one.

It's your move

Grab a piece of the future for yourself. Visit your local computer store. We'll give you the address of the Apple dealer nearest you when you call our toll-free number. Then drop by and sink your teeth into an Apple. **800-538-9696**. In California, **800-662-9238**.

CIRCLE INQUIRY NO. 3

apple computer

in the operation of the business. The system is used in payroll, inventory, engineering, planning, and even in the testing phase of the products. This is something not many computer companies can boast.

Around the Hayward area is Mullen Computer Products, 2306 American Avenue, #6, Hayward, CA 94545, phone (415) 783-2866, Attn: Bob Mullen. Bob and his crew came up with something that every Heath H8 owner will want right now; a bus extender board, shown in Photo 6. This board sells for $39 and is available either directly from Mullen or Godbout or any of the computer stores that carry Godbout product lines.

While up in the 'Gulch' I had the chance to visit Dragon country — Menlo Park. That's where Bob Albrecht and his team of writers call home. As I was roaming around the city I ran across a very interesting book store that was catering to the computer aficionados in the area. From Photos 7 and 8 you can see that they have taken an interesting approach to catching the eye of just about everyone. This store is Keppler's Bookstore, located at 825 El Camino Real, Menlo Park, CA 94025, phone (415) 324-4321. Besides having computer games available to the customers, they carry one of the most complete lines of computer books that I have seen today.

PHOTO 7

PHOTO 8

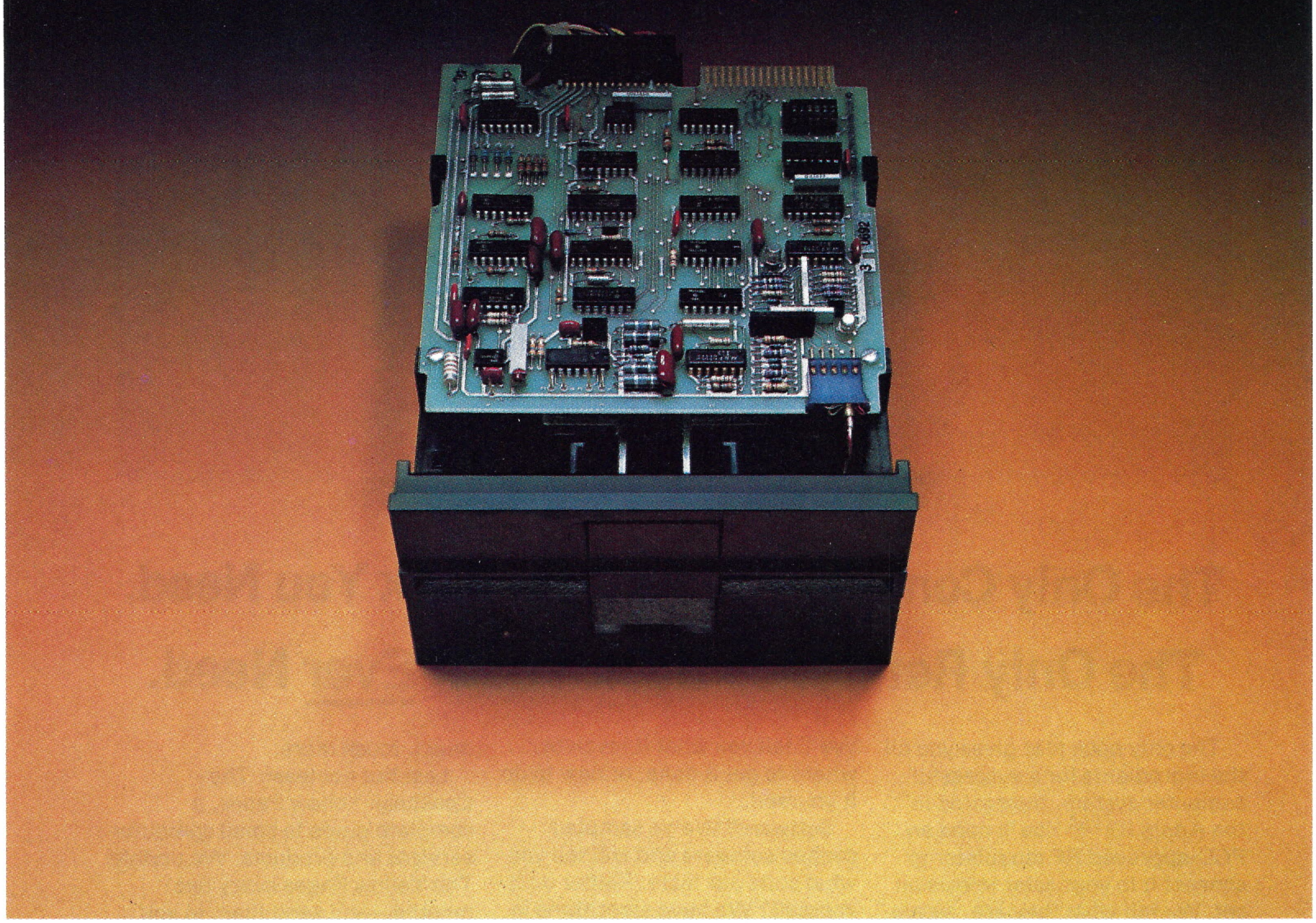
SOME OTHER NEWS

We just recently learned that Billings Computer has purchased the disk division of Calcomp and will be renaming it CalDisk. This they feel will greatly enhance their product line and make it possible to provide more customer support.

Some of you have been wondering about Summagraphics, 35 Brentwood Avenue, Fairfield, CT 06430, (203) 384-1344, and their $1,000 offer for authors who write articles on their Bit Pad™ being true. I can testify that it is in fact real.

Recently, we purchased an article from Marvin Mallon on an application for the Bit Pad and told him to advise Summagraphics. Within a few short days after sending them his letter and a copy of our contract, Marvin was pleasantly surprised with a check for $1,000.

If it isn't Shugart, it isn't minifloppy.™



Shugart invented the minifloppy in 1976. Today there are more than 100,000 of the little drives in use. That's because users want the affordable random access data storage of the minifloppy.

Shugart packs years of proven floppy drive technology into this tiny package. Up to 110 kbytes of data storage. Fast random access of about one-half second. And high speed data transfer of 125 kbits per second. Plus sensible, maintenance-free features like write protect to prevent accidental data loss, an activity light to indicate when the drive is selected by your computer and a door interlock to protect your media from damage.

Our proprietary read/write head provides maximum data interchange margins, and it is positioned precisely on the selected track by a patented spiral cam actuator. The DC drive motor with integral tachometer assures accurate diskette rotation and low heat dissipation. A die cast aluminum base plate provides a solid foundation for the drive.

At Shugart, technology leadership is more than a slogan, it's a commitment. Get reliability and value when you invest your money for floppy disk storage. Ask for the standard of the industry. minifloppy. If it isn't Shugart, it isn't minifloppy.

Shugart
435 Oakmead Parkway, Sunnyvale, California 94086

See opposite page for list of manufacturers featuring Shugart's minifloppy in their systems.

TM minifloppy is a registered trademark of Shugart Associates

So you can see some companies really do live up to their promises. The article is a good one and will be in next month's issue.

INTERESTING PUBLICATIONS

The other day I received a copy of the TRS YELLOW PAGE, 96 Dothan Street, Arlington, MA 02174 which is a short ad catalog of TRS-80 software produced by the Micro Architect apparently at the same address. This is a handy little guide and doesn't cost anything unless you want to advertise in it.

An adversary friend of mine, Fred Gruenberger, sent me a copy of his magazine POPULAR COMPUTING, which is the second oldest magazine in the microcomputer field. This magazine is about a twenty-page resource document for the true computer scientist. The monthly publication is available for $17.50 a year from POPULAR COMPUTING, Box 272, Calabasa, CA 91302. Should you have the need to have number systems and classic algorithms explained, this is the magazine you need. Fred had been in the computer business long before most of us were born, and has a lot of knowledge to share, which he does quite ably each month. Fred also keeps an eye on publications such as INTERFACE AGE and advises us on our strong and weak points. We don't necessarily always agree, but I can assure you that I listen very closely when Fred takes the time to call me.

There is another magazine that some of you may take a little umbrage with me for mentioning. The magazine called FUSION, the magazine of the Fusion Engergy Foundation. Subscription is $18 for ten issues from the Fusion Energy Foundation, P.O. Box 1943, New York, NY 10001.

The purpose of FUSION is to explain nuclear energy and try to take away some of the myths around it. The magazine comes out totally in favor of nuclear as a power source. Due to the recent publicity surrounding nuclear power plants, you may be against this form of energy development. However, the magazine does offer some extremely in-depth articles on how reactors work and explains plasmas. Technology buffs will enjoy the magazine for its technical content and not necessarily for its political satire.

THE LETTERS I GET

Lately I have been getting a number of complaint letters from readers teling me war stories about companies that they have dealt with. I immediately send a copy of this letter to the company in question and suggest that they work out some kind of settlement with the writer. This has been working fairly well, and in just about all instances the companies have provided the person with some sort of just settlement.

However, a problem I warned you about several months ago has been cropping up. The problem is of generating a complaint letter before all the facts are known. This has happened in ten cases. The writer quickly sent us another letter telling the problem had been resolved.

What is of utmost importance is that you have your facts straight and are sure than an injustice has been done. Then document all letters, phone calls and visits. Know who you talked to and mark down the date and time. Keep a special file that is easily referenced; in fact, use your computer as the reference source. But be sure you have taken every step to cure the problem before writing to a magazine. We will try to help, but can only do so when we are satisfied that every attempt possible has been made by you to achieve satisfaction.

IN THE MAKING

We have some plans that should make reading INTERFACE AGE even more enjoyable and informative. One of these plans is the incorporation of a new section in the magazine entitled: THE LEARNING CENTER. This section will bring to you such things as learning to program on the TRS-80, using the computer in the school. How children can make the most of the home computer. Educational games for such systems as the APPLE, and Atari computers. If that isn't enough, we still have some more things up our sleeve that you will see introduced in the pages of INTERFACE AGE in the next few months.

Also the first volume of INTERFACE AGE Golden Classics is in the making and should be ready sometime in the early fall. This first volume will be on software and will contain those programs that many of you have asked to see again.

In line with all these plans and changes, Bob Albrecht wants me to tell everyone that COMPUTER TOWN USA is coming, and if you want to find out more about it write to: Computer Town U.S.A., P.O. Box 310, Menlo Park, CA 94025. This is one of the most unique enterprises ever dreamed up — even by Dragons — that the industry has seen to date.

THE TRADES DO IT AGAIN

In a March issue of *Electronic News,* a front page article was published that presented a list of all the microcomputer compnies that are currently in trouble or have already gone under Chapter 11 protection. The article seemed to present an aura of a sour industry, something the large tabloids have a penchant for doing.

The fact is that the microcomputer industry is very much alive and doing very well. Sales are escalating beyond anyone's dreams and large companies such as Hewlett Packard, Texas Instruments, Mattel, and Atari are heavily engaged in the support of the industry and in enlarging the market. This would definitely not be happening if the marketplace was falling apart.

Unfortunately, journalists working for large trade tabloids like to look on the bad side of things and sensationalize them. Possibly this is because the large mainframers have supported them for so long and it is to their advantage to try to tear down a rapidly growing industry.

The microcomputer industry is taking some of the share of the market away from the mainframers, but the pie is extremely large and can support both ends of the industry.

—carl

LETTERS TO THE EDITOR

ENJOYING NTS

Dear Editor:

I'm glad to see that INTERFACE AGE has the initiative to print the NTS mini-series. Some of it may be review but for me it's the first time all the pieces will be interfaced with my background in math. It doesn't hurt to have a good strong foundation in the basics before going on to the application.

I don't have a computer but have been interested in the mechanics, programs and applications in various fields for years. Your magazine covers a wide range of topics but with sufficient detail. It leaves room for thought.

I look forward to the rest of the NTS series and the other tutorial programs that you will offer. Keep up the good work!

Noreen Kerr
Buffalo, NY

Noreen's letter is representative of the 150 letters we have received from female readers. We have also received a number of letters which were answered directly regarding splitting the units and providing answers.

We are making every effort to avoid splitting any given unit and NTS is providing answer sheets with mailed cards. Also we had a number of complaints regarding the speed that cards were being answered. This was our fault. In the last several weeks we have taken steps to speed up the answering process.

UNDERSTANDING SPECTRAL MUSIC

Dear Editor:

G.S. Stiles in his article, "Spectral Music" (March, 1979) uses the concept of musical spectrum in a seriously misleading way. We typically speak of the spectrum of *sound waves* when referring to musical spectra. Stiles should have made it absolutely clear that he was using this term in a highly unconventional way.

There is an incredible difference between finding the spectrum of sound waves (an important scientific technique) and that of applying the technique to arbitrarily encoded music notation symbols (as Stiles suggests). His remarks about good music possessing a characteristic spectrum suggests Stiles himself might be confused here — it's the sound waves that tend to have characteristic spectra.

Please refer to the following references for applications of spectral analysis of music using computers.

Piszczalski and Galler. "Automatic Music Transcription," *Computer* Music Journal 1 (4), 1978.

Backus, John. *The Acoustic Foundations of Music.* Norton, 1969. p. 100.

Martin Piszczalski
Research Associate
University of Michigan
Department of Computer and Communication Sciences
105 South State, 2076 Frieze Bldg.
Ann Arbor, MI 48104

ALMOST HAPPY!

Dear Editor:

Please be so kind as to let me share with you some thoughts on my recent subscription to INTERFACE AGE as it rounds the half-way mark.

1. I am disappointed in the quantity of software. Under no circumstances should you sacrifice quality, but we'll get to that later. I particularly desire assembler language programs for Z-80. Perhaps I'm suffering from misconceptions, but I believe that the Z-80 has captured a substantial share of the market. I personally have finally reached the stage where, at long last, I am getting some Z-80 programs running, as are some immediate associates. The lack of source programs hasn't made the job any easier. Do you have any opinions on the marketability of these through your magazine? Count me one pro vote.
2. Back to quality. I've noticed a considerable number of typo errors in your general reading (and some ads!). This could serve only to deteriorate interest in your publication. . . especially if it slips into your software articles.

Contrary to my presentation, please don't think of me as a critic (wholly). I only wrote because I *almost* enjoy YOUR magazine!

Charles W. Butler
Lansing, MI

Charles, last November we made a decision to make our software readable all the time. This meant the layout had to be horizontal instead of vertical to the page, which takes more space. Even by reducing the total number of software pages printed per issue, we still print more than anyone else. We also provide more assembly, 8080, Z-80, 6800 programs than the other books.

You will also notice that we are providing a fair amount of support to the TRS-80. Where we have a problem is for the Apple, Heath H-11, Bally and Atari machines. We would love to publish articles, particularly software, around these machines, but apparently the users of these machines have no inclination to write.

INTERFACE AGE INDEX

Dear Editor:

I am interested in a yearly index of articles that you have printed since 1976. Could you help me?

Thomas Brey
East Hartford, CT

Tom, if you check the March 1978 and January 1979 issues you will find what you need.

It's already a big success!

The party's over for all dumb terminals and a lot of smart ones too. But, at $798 (quantity 25), the party's just beginning for Intertec's InterTube II.

Standard features to celebrate include a full 24 line by 80 character display, 128 upper and lower case ASCII characters, reverse video, complete cursor addressing and control, an 18 key numeric keypad, special function keys, blinking, protected fields, character and line insert/delete, editing, eleven special graphics symbols, a 25th status line which displays the terminal operating mode and an RS-232 printer port.

* $995 Quantity One

You'll discover even more reasons to celebrate when you sit down in front of an InterTube II. Our wide bandwidth monitor produces crisp, sharp chararters everywhere on the screen. InterTube's Z-80 processor enables a host of operator oriented features to boost the efficiency of both software and programmers. And, InterTube's rugged modular design combined with its built-in self-test mode insures quick and reliable servicing.

InterTube's price/performance ratio can satisfy your requirements whether they be a sophisticated data entry application or a simple inquiry/response environment. So, there's really no reason to think "dumb" when you can afford to be so smart!

Join the thousands of InterTube celebrations going on around the country at this very moment. Call us at the number below and start your own celebration (BYOB—we'll bring the InterTube).

INTERTEC DATA SYSTEMS®

2300 Broad River Road, Columbia, S. C. 29210
(803) 798-9100 TWX: 810-666-2115

CIRCLE INQUIRY NO. 36

the Column

By Michael P. Shepanski

FOUR EXCITING NEW DEVELOPMENTS FOR TRS-80 USERS WITH LEVEL 1

INTRODUCTION

Until now, the following facilities have been lying dormant in Level 1 BASIC:

1. Six new arithmetic operators associated with the six relational operators.
2. Four new logic operators: XOR (exclusive OR), EQV (equivalence function), IMP (conditional operator), NOT (negation).
3. Multiple INPUT statements — ability to assign values to two or more variables in one INPUT statement.
4. The ability to play pre-recorded vocal messages (or other sounds) at a particular point in a program or to record sounds (e.g., from a radio) at a pre-set time.

No longer need the TRS-80 users and programmers be ignorant of these facilities, as this article will show.

RELATIONAL OPERATORS VS ARITHMETIC OPERATORS

The first thing to be understood is that the relational operators (>,<,>=,<=,=,<>) are, in fact, arithmetic operators. If you do not believe this, try the following calculator-mode statement:

```
PRINT 3<4
```

Press "ENTER", and your machine will print 1! Why doesn't the machine put:

```
WHAT?
READY
>_
```

on the screen instead of "1"? What has happened is that the computer has used the expression from Table 1 corresponding to the < operator (in this case, "INT (SGN(B-A)/2 + .5)") to come up with the answer (viz. 1).

Try all the relational operators in this way. Try using different numbers. Can you see a pattern? Have you tried this one?

```
PRINT 3 = 1
```

Here your computer should reply with a "0" (using the expression "1 - SGN (ABS(A-B))" from Table 1).

Table 1.

Expression	Formula
A > B	INT(SGN†(A−B)/2+.5)
A < B	INT(SGN†(B−A)/2+.5)
A >= B	INT(SGN†(A−B)/2+1)
A <= B	INT(SGN†(B−A)/2+1)
A = B	1−(SGN†(ABS(A−B))
A >< B	(SGN†(ABS(A−B))

†The SGN function is not available directly in Level 1 but may be simply defined as being 1 when its argument is positive, −1 when it is negative and 0 when it is 0.

CIRCLE INQUIRY NO. 20

You have probably noticed that the expressions which look like true statements (e.g. "3<4") will always give you a "1", and those which look like falsehoods (e.g. "3=4") will give you a "0".

Try some more examples. Try with arithmetic operators or with variables.

Now you will see that these are arithmetic operators, but they are a very special type, and they are useful, too. If you wanted to find the SGN of a number (call it X), you could use the subroutine from the back of the book, but that uses 87 bytes and doesn't show much imagination. Besides, you can't access it halfway through an arithmetic expression. But with our six new arithmetic operators, we can get the SGN of X by this expression:

(X>0) — (X<0)

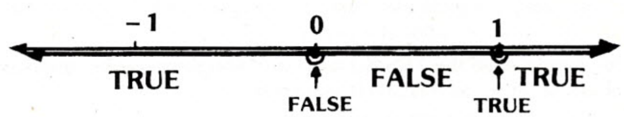
IF-THEN — HOW DOES IT WORK?

Now type in this little program:

```
10 INPUT A
20 IF A THEN PRINT "YES"
```

and RUN for different values of A. How can what you see be explained? We know that all the relational operators return numbers (1's and 0's) so the antecedent (the part between the IF and THEN) must always be evaluated to give a number. It is really no great surprise that the machine needs a number for the antecedent.

There is the problem of which antecedent — numbers the TRS-80 takes as "true" and which ones it takes as "false." In the above program, you should find that values of A from 1 up, as well as any negative ones, count as "true," whereas values from 0 to 0.999999 inclusive count as "false," as shown in the number-line below:



If you had this statement in the program:

```
60 IF 4<5 THEN 420
```

then the TRS-80 would evaluate the arithmetic expression "4<5" according to the appropriate expression from Table 1. This returns a 1, and as we saw in the program we entered earlier, a 1 counts as true, so processing is transferred to line 420. However, in the example:

```
60 IF 4>5 THEN 420
```

it will evaluate the expression "4>5" according to the appropriate expression from Table 1. However, this time it returns a 0 which counts as false, so processing does not transfer to line 420. Instead it goes on to the next statement in line-number order.

LOGIC OPERATORS VS ARITHMETIC OPERATORS

We may well ask, what do the "*" and "+" logic operators do to their operands? For instance, in the statement:

```
60 IF (4<5)*(4>5) THEN 240
```

what does the * do? It simply multiplies. The computer does the parentheses first, of course, leaving:

```
60 IF 1*0 THEN 240
```

and then it does the multiplication:

```
60 IF 0 THEN 240
```

which leaves the antecedent as 0, which means it is false.

If we replace the * in 60 by a +, we have:

```
60 IF (4<5)+(4>5) THEN 240
```

Evaluating the parentheses, we have:

```
60 IF 1+0 THEN 240
```

that is:

```
60 IF 1 THEN 240
```

which counts as true.

In the above two examples, the * worked well for "AND", and the + worked well for "OR". But do they always?

First we will look at the use of multiplication for AND. The truth table for the AND function is:

A AND B	A = T	A = F
B = T	T	F
B = F	F	F

and assuming we only have operands which are 1 or 0, the multiplication table is:

A * B	A = 1	A = 0
B = 1	1	0
B = 0	0	0

If you compare the two tables, it is easy to see why the multiplication works for AND. It returns a 1 if both A and B are 1. Otherwise it returns a 0, which is convenient because 1 stands for truth and 0 for falsehood.

Now we must address the use of addition for OR. First we will look at the truth table for the OR function:

A OR B	A = T	A = F
B = T	T	T
B = F	T	F

and, again, assuming we have operands of only 1 and 0, the addition table is:

A + B	A = 1	A = 0
B = 1	2	1
B = 0	1	0

The strategy here is very much the same as for the use of multiplication table, i.e. there is a 0 corresponding to a false result and a 1 corresponding to a true result except in the cases where both operands equal 1, which makes the result of the operation equal to 2. To prove that it does, enter this calculator-mode statement:

```
PRINT (5<7)+(9>=4)
```

You will see that the expression is evaluated to give 2. No all "true statements" have a value of 1, but this does not matter. You will remember that all antecedents with a value of 1 or over (as well as the negative ones) count as true. If the antecedent is 2, it will not make any difference.

What if this result is used as an operand of another + function? The result of that + function will be a 3. If this is used in another + function, the result will be a 4. This could go on forever, and it would make no difference because any of these numbers as an antecedent will pass as true. Furthermore, some reflection will reveal that using a 2, 3, 4 or some other such number as an operand in a * function will not disturb anything either. That is why * and + make such good analogs of AND and OR, respectively.

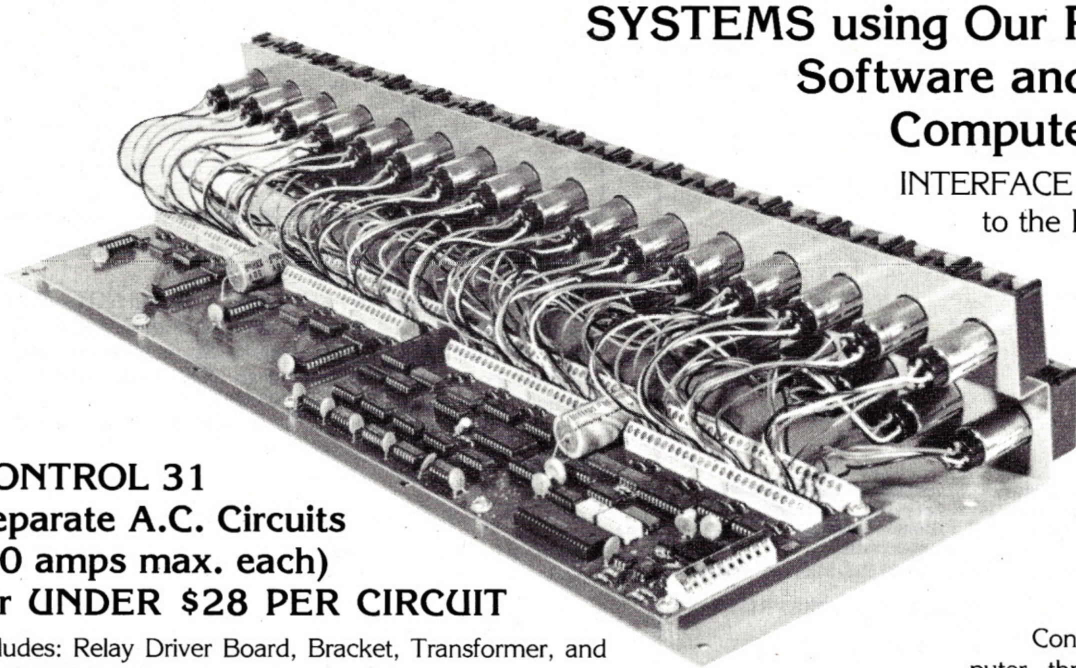
Consequently, we know that all comparisons and antecedents are just numbers, that logic operators are just arithmetic operators which work on those numbers and that we have six new arithmetic operators.

LOGIC OPERATORS UNLIMITED

We now have a total of ten arithmetic operators (four old, six new), and we are only using two of them (* and +) as logic operators. We could use all of them, but most are fairly useless. However, some

GHOST POWER

BOARDS for ALL COMPUTERS, or COMPLETE TURNKEY SYSTEMS using Our Real Time Software and GHOST Computer System



INTERFACE your Computer to the Real World with **GIMIX Relay Driver Boards.**

CONTROL 31 Separate A.C. Circuits (20 amps max. each) for UNDER $28 PER CIRCUIT

Includes: Relay Driver Board, Bracket, Transformer, and 31 GE RR-8 Relays. Assembled and Tested with Documentation and Software Driver Routines. **$\$848^{05}$**

Relay Driver Boards Only **$\$448^{86}$**

Connects to any Computer through a 20 ma. current loop (up to 4 Boards 128 Relays per port.) Reports the on/off status of each relay on demand.

INTERFACE the **Real World** to your **Computer** with **GIMIX OPTO-BOARDS and REMOTE KEYPAD SYSTEMS:**

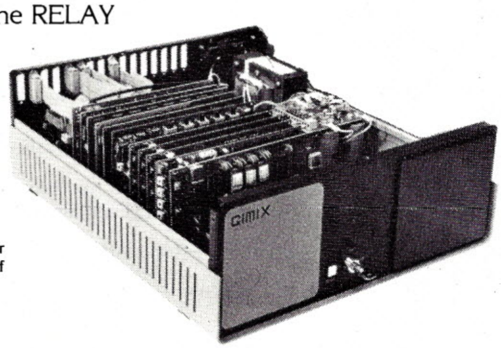
OPTO-BOARDS Connect One 8 Bit Parallel I/O Port to as many as 34 Outside World Switch Closures (e.g. doorbells, eyes) with: 1500 Volt Isolation, built-in contact debounce, on board scanning and buffer.

REMOTE KEYPAD SYSTEMS

Provide Remote Data Entry for Computer Systems or Directly Control One RELAY DRIVER BOARD or STEREO SYSTEM CONTROLLER.

6800-SS 50 BUS 16K SYSTEMS

from **$\$1294^{29}$**



Includes: Mainframe cabinet, mother board, power supply, fan, CPU, 16K static RAM, and choice of I/O card.

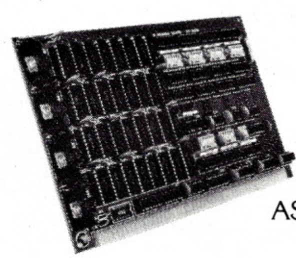
Other packages available.

16K Static RAM Boards for the SS-50 Bus

MORE FEATURES:
- Gold bus connectors
- 4 separate 4K Blocks
- Individual Addressing Write Protect, and Enable/Disable for each Block
- Tested at 2MHz

AT A LOWER PRICE

ASSEMBLED **$\$298^{13}$**



As above with Sockets and Software control features.

ASSEMBLED **$\$368^{16}$**

THAT'S VALUE!

We would like to call your attention to the **June 1979** issue of **Interface Age Magazine.** It will contain an article about **GIMIX** and go into more detail about our **GHOST Power Control System** than we can in this ad.

GIMIX INC.

The Company that delivers. *Quality Electronic products since 1975.*

1337 WEST 37th PLACE • CHICAGO, ILLINOIS 60609
(312) 927-5510 • TWX 910-221-4055

Table 2.

Logical Formula	Level 1 Equivalent
A AND B	A * B
A OR B	A + B > 0
A XOR B	A < > B
A EQV B	A = B
A IMP B	A <= B
NOT A	1 − A

come in very handy. Some useful ones are listed in Table 2, but you can make up some more yourself.

NOTE: The operators for XOR, EQV, IMP and NOT require that the operands be either 1 or 0, so Table 2 includes a new form of OR which, although slightly more complicated, is necessary whenever the result becomes an operand of an XOR, EQV, IMP, or NOT function. Otherwise, use the old + for OR.

If you wanted to print X if A = 3 or B = 6, but not both, you would use a statement like this:

```
1461  IF (A=3) <> (B=6) THEN PRINT X
```

If you are interested in memory conservation, you can contract it to this:

```
1461IF(A=3)<>(B=6)P.X
```

but you can't take away the parentheses. Even though >,<,>=,<=,= and <> are arithmetic operators, they have no order of operations. That means that if you have the statement:

```
1461IFA=3<>B=6P.X
```

the computer might give you a right answer, it might give you a wrong answer, but most likely it will print the depressing message:

```
WHAT?
1461  IFA=3?<>B=6P.X
```

Keep the parentheses, and make sure that you never have two relational operators like this:

```
A < B > = C
```

when what you mean is:

```
(A < B)*(B>=C)
```

For another example, say you want to clear the screen if it is not the case that either A or B are equal to 5. Obviously, there are a few ways to approach this, but the most direct is:

```
8691  IF 1 — ((A=5) + (B=5) <> 0) THEN CLS
```

Notice that we use the new form of OR because the result is the operand of a NOT function. Of course, you could abbreviate this to:

```
8691IF1—((A=5)+(B=5)<>0)C.
```

Though "C." is not in the back of the book as an abbreviation, it works.

Finally, it should be pointed out that there is no limit to the fooling around you can do in antecedents using the ten arithmetic operators and the two arithmetic functions "ABS" and "INT". You might even find a use for "RND" somewhere. Also, the relational operators are invaluable in arithmetical expressions.

MULTIPLE INPUT STATEMENTS

As you undoubtedly know already, Level 1 BASIC permits the statement:

```
120  INPUT X
```

 CIRCLE INQUIRY NO. 35

or, if you wish:

120 INPUT "WHAT IS THE X-COORDINATE"; X

Now, it would appear that, to input two numbers, two statements must be used, like this:

120 INPUT X; INPUT Y

or, in a more informative form:

120 INPUT "WHAT IS THE X-COORDINATE"; X
130 INPUT "WHAT IS THE Y-COORDINATE"; Y

However, it is possible (although it is not explained in the manual) to input values for both variables in a single statement, like this:

120 INPUT X, Y

or like this:

120 INPUT "WHAT ARE THE X AND Y COORDINATES"; X,Y

When these examples are run, there will be no indication that they are multiple-input statements, except in the printed messages. For example, there is just a "?" printed on the display. The user must then enter two numbers (or expressions) with a comma (",") between, like this:

WHAT ARE THE X AND Y CORODINATES?2,5

Then when the user presses 'ENTER', the two values are assigned to the two variables, respectively. To prove this, type in this program:

10 INPUT A,B
20 PRINT A*B, A+B, A/B, A-B

and RUN, remembering to enter the two values with commas between. When you enter only one number, change line 10 to read:

10 INPUT "ENTER TWO NUMBERS WITH A COMMA BETWEEN THEM"; A,B

and RUN again. Now the person at the keyboard cannot go wrong. But, just out of interest, try entering three numbers with commas between. The computer ignores the extra number.

This facility is not limited to only two variables. You can use the whole alphabet and even some more of the array. It doesn't make any difference, as long as the person at the keyboard knows to use a comma and the right number of values.

You may have noticed that in single string input lines, if you input a string with a comma in it, the comma and everything after it will be ignored. The reason here is simple; the TRS-80 interprets this as a multiple input, and since the INPUT statement only asked for one string, the second string is ignored. Thus, if in the program:

10 INPUT "WHAT IS YOUR NAME";A$
20 PRINT "YOUR NAME IS";A$

the user enters "MERVIN,THE MAGNIFICENT", the computer will ignore both the comma and "THE MAGNIFICENT".

COMPUTER CONTROL OF TAPE RECORDER

By now you should be aware of the TRS-80's ability to store and load programs or data on audio cassettes with the built-in cassette tape interface. You may already know that the EAR plug is used for the transfer of data during CLOAD and INPUT#. You may also know that the AUX plug is used for the transfer of data during CSAVE and PRINT#.

What is important, however, is the function of the REM plug. It carries a signal which enables or disables the tape recorder motor. It enables the motor to operate as long as necessary during CSAVE, CLOAD and INPUT#, but during PRINT# the motor is enabled for a fixed time of between 4 and 5 seconds. If the REM and AUX plugs are disconnected from their jacks, the tape recorder motor may be started for that period. Thus, if you record a message, rewind the tape to the beginning of that message and press the "PLAY" button with the REM plug connected, then should the computer come across a PRINT# instruction either from the keyboard or in a program, the tape recorder will be activated for 4 to 5 seconds, and the recorded message (or the first 4 to 5 seconds thereof) will be played. To play longer messages, use two or more PRINT# instructions.

Recording can be done still using the PRINT# instruction by pressing the "RECORD" button and "PLAY" button together instead of only pressing the "PLAY" button. Then when a PRINT# instruction is encountered, 4 to 5 seconds of sound will be recorded through the CTR-41's condenser microphone (you could use an external microphone, but most have a plug which goes in the REM jack which is already occupied by the TRS-80's REM plus).

For instance, if you are leaving the house at 9 a.m. and you want to listen to a radio broadcast which starts at 10 a.m. and continues until 10:30, before you leave the house in the morning you should do the following:

First, turn on your radio and tune it in to the required frequency.

Second, program your computer (you may have to bend the rules and turn off the radio temporarily at this point) to wait one hour, using a series of timer loops (since the integer limit is 32767, FOR-NEXT loops have an upper limit of 32766, which takes just over a minute on the author's system, so you should use nested timer loops for long periods). At the end of that hour, the computer is programmed to activate the tape recorder for half an hour (this can be accomplished by a FOR-NEXT loop enclosing a PRINT# instruction — since each PRINT# instruction records for 4 to 5 seconds depending on the system, to record for half an hour. The FOR-NEXT loop will have to go through, very roughly, 4 000 iterations, all depending on the particular TRS-80).

Third, put your cassette in the recorder and bring it to the position at which you wish to begin recording, then connect the REM plug (but no others) and press PLAY.

Finally, just before you go out the door, type in RUN and press ENTER. On return, you will find that the broadcast has been recorded.

SUMMARY

None of the above features of the TRS-80 are mentioned in the User's Manual. The extended use of operators is, no doubt made possible only by the quirks of the Level 1 interpreter, and the ability to play and record sounds results from the technical arrangement of the cassette recorder and its interface. On the other hand, it seems that the manual's failure to document multiple-input statements may have been an oversight.□

CHICAGO ELECTRONICS SHOW

The spring Consumer Electronics Show will be held in Chicago, June 3-6. The show demonstrates many types of electronics available or soon to be available for the consumer, ranging from stereos to computers.

For more information contact the Consumer Electronics Show, Two Illinois Center, Suite 1607, 233 N. Michigan, Chicago, IL 60601.

METRIC STANDARDS STRATEGY MEETING SCHEDULED

"Developing the U.S. Metric Standards Strategy" is the topic of a seminar to be held June 4-5 at the Sheraton O'Hare Motor Hotel in Chicago. The American National Standards Institute, which coordinates the development of voluntary national standards in the United States, is sponsoring the meeting.

Registration information is available from Claude H. Burns, deputy managing director, American National Standards Institute, 1430 Broadway, New York, NY 10018.

REGIONAL DP CONFERENCE

The Sacramento Chapter of the Data Processing Management Association is hosting the Region 2 conference in Sacramento, California, June 10-12. The theme of the event is "A Technological Gateway to the Eighties." Tickets are $85 for members, $95 for non-members.

For details contact Sam Price, Data Processing Management Association, P.O. Box 1223, Sacramento, CA 95806.

INFORMATION RESOURCE FORUM

The Society for Management Information Systems is holding a "Forum on Information Resource Management" at the Drake Hotel in Chicago on June 25-26. Addressing the theme of "Information Resource Management in the Years of Change," the Forum will provide an outlook for the next decade in terms of information systems technology.

Sessions and speakers will provide a management update in areas such as auditing and security, impact of distributed processing and data base management.

A key feature of the Forum will be an outlook panel of top MIS directors representing various industries who will present a perspective on the problems and potential of information resource management in their organizations and industries.

For details contact Ken Burroughs, DBD Systems, 1500 N. Beauregard St., Alexandria, VA 22311, (703) 820-3310.

CHORAFAS SEMINAR

"Distributed Information Systems" will be the topic of two identical seminars scheduled for June, 1979 by Dr. Dimitris N. Chorafas, internationally-known management and data processing consultant.

The first will be held June 11-15 at the Radisson-Chicago Hotel, Chicago; the second June 18-22 at the Breckenridge Pavilion Hotel, St. Louis. Topics include new communications technologies; use of protocols and data bases; new software developments and applications.

For details contact Richard Laubhan, Project Communications Inc., 333 E. Ontario, Suite 2603B, Chicago, IL 60611, (312) 266-2113.

ADVANCED SEMINAR OFFERED

Polytechnic Institute of New York and the Institute for Advanced Professional Studies are presenting a seminar, directed by Dr. Glen Marston, titled "Testing Microprocessor-Based Systems." The workshop enables design and test engineers to coordinate microprocessor-based product design with production testing.

The seminar will be held June 11-15 at The Colonial Hilton Inn, Wakefield, Massachusetts. Tuition is $495. For details contact Prof. Donald D. French at (617) 964-1412 or the Institute for Advanced Professional Studies, One Gateway Cir., Newton, MA 02158.

DATA ENTRY COURSE

Management Information Corporation is sponsoring an interactive seminar that deals with data entry. Discussions with instructors and other participants will provide solutions to problems that can be easily implemented.

"Data Entry Management and Supervision Seminar" includes instruction in data entry system concepts, organization of the data entry department, data entry control techniques and operator training.

The course will be at the Cherry Hill Inn in Cherry Hill, New Jersey on June 18-20.

For fees and details contact Management Information Corp., 140 Barclay Center, Cherry Hill, NJ 08034, (609) 428-1020.

TWO-DAY MANAGEMENT SEMINAR

The cash impact of manufacturing decisions is the focus of "Effective Production Planning and Inventory Management," a seminar presented by the Wharton School of the University of Pennsylvania on June 25-26 in Chicago.

Designed for manufacturing organization executives, the seminar presents techniques to increase working capital, reduce inventories without compromising service, forecast market demands, resolve potential problems and minimize shortages and production delays.

For more information contact Heidi E. Kaplan, Department 20 NR, New York Management Center, 360 Lexington Ave., New York, NY 10017, (212) 953-7262.

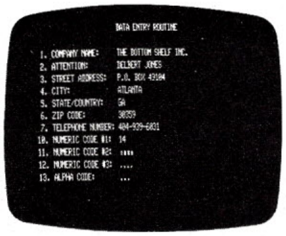
COMPUTER SWAP MEET

The North Orange County Computer Club will be holding a computer and electronics swap meet Sunday, June 17. Buyers are admitted free. The sale will be held at Advanced Computer Products, 1310-B E. Edinger, Santa Ana, California.

 CIRCLE INQUIRY NO. 93

TBS OFFERS COMPLETE SYSTEMS BACKUP.

Mail List — $125.00



For your 32K, two disk drive and printer TRS-80, an endless mailing list!

MAIL LIST keeps 500 names and addresses per disk, together with three numeric codes and one alphabetic code for each name. 500 records per disk, and any number of disks that you wish. MAIL LIST keeps all entries in zip code and then last name order, and keeps track of which disk is holding which items. Automatic prompts instruct the operator as to which disk should be inserted. Selectable printouts for labels or other lists for hard copy storage. Full EDIT, DELETE, and other commands for your operators convenience. See SYSTEMS EXTENSIONS or your dealer for full details. **The first TRS-80 batch-processing program.**

Diskettes — Box of 10, $38.00

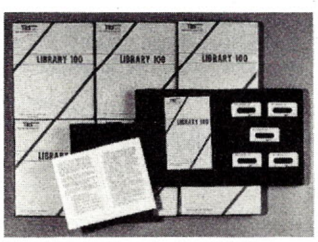
New from TBS and Wabash Tape Corp. TBS, the leading name in TRS-80 software, and Wabash, the leading name in computer magnetic recording materials, presents, for your microcomputer disk system 5¼″ mini diskettes.

Only TBS and Wabash can assure you that each diskette has been individually certified, and certified to 3200 BPI! Further, EACH one is guaranteed for one year after purchase.

Cassettes — $1.90 3 for $5.40

Our cassettes are designed specially for use in RADIO SHACKS'S TRS-80 and other audio based computers. TBS backs its cassettes with a 90 day warranty against parts and material defects. TBS has equipped its cassettes with special features for you: sliding write protect tabs, extra large pressure pad and a 5 screw assembly. These cassettes are of TBS quality, built for your convenience in your TRS-80 or other audio based computers.

Library 100 — $49.50

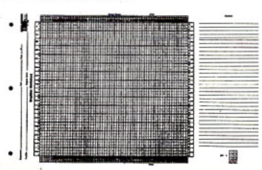


The Library 100 was designed to fulfill your general TRS-80 Level II programming needs. Using advanced Level II techniques and rigid quality control, The Bottom Shelf has been able to make available 100 programs on five guaranteed-to-load cassettes, which load over a wide range due to advanced recording techniques and methods designed by engineers for The Bottom Shelf. The programs include applications in five areas: Business and Finance, Education, Graphics, Home and Games.

As an added bonus, you get Tiny PILOT, the first new high-level language for the TRS-80. It's perfect for teachers, parents, students and sales trainees.

Using only six of Tiny PILOT's commands, even a child can program in minutes.

TBS Pad System — $19.95 set of five



Programmers pads, 11½″ x 17″. Punch for insertion and storage in a standard three ring notebook binder. All necessary indexes and numbers for your programming convenience on a 16 x 64 CRT.

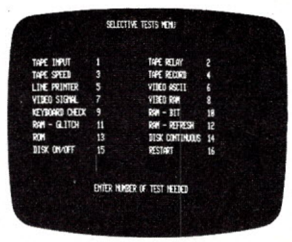
Systems Extensions — $3.00

Systems Extensions is published and marketed by TBS, creators of the LIBRARY 100. The articles published in Systems Extensions were written by our staff of TOP QUALITY PRO-GRAMMERS at TBS, to aid you with your computer. Also incorporated in this publication is a group of over 300 items designed to support your computer system.

Partial Table Of Contents

1. Computers of the Past
2. Computers the Present and Next Two Years
3. Computers the Future and Next Ten Years
4. TRS-80 and the Business System Community
5. Standards for Professional Programming
6. Preparing for Programming A Business and Professional Application
7. Methods to Program Your System
8. Review of the Electric Pencil
9. Random Ramblings
10. Computer Aided Instruction
11. The Diskette Revolution
12. Level II Index
13. The Purchase, Care and Maintenance of A Business Computer
14. Your Computer and the Wall Socket

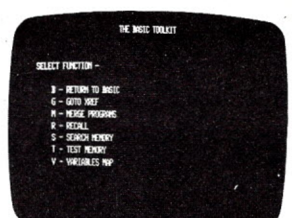
System Doctor — $28.50



Now you can check your TRS-80 Level II computer and all it's peripherals with this program. SYSTEM DOCTOR allows you to selectively test any function or peripheral or continuously test your system. Run this program once a week or when trouble occurs. Be your own computer diagnostician.

The Basic Toolkit — $19.80

By F. Barry Mulligan



A Programmer's Solution. The Basic Toolkit is a dream come true! This machine language program will do the following:

- Search a basic program, and print out to screen or printer, an alphabetized listing of the variables used in the program and the line numbers in which they are found.
- Search and print to screen or printer, a listing of all GOTO's and GOSUB's and the line numbers in which they appear.
- Restore basic programs that have been accidentally lost (after typing NEW, or WHEN YOU GO TO DOS.)
- Will check for bad memory in 15 seconds.
- WILL MERGE PROGRAMS ON A CASSETTE BASED SYSTEM!
- Will search memory for all occurences of a specific byte and list the locations where it appeared.

This program can be accessed at any time while you are programming by hitting SHIFT/BREAK.

Be a participating TBS dealer. Phone: 404-939-6031

See your local TBS dealer, over 100 nationwide. Master Charge or Visa accepted. All orders plus $2.00 P & H. Add 4% tax, if Georgia. Above software for TRS-80 L II. **CIRCLE INQUIRY NO. 9**

TBS INC.

The Bottom Shelf, Inc.
P.O. Box 49104
Atlanta, Georgia 30359
Phone: 404-939-6031

COMPUTERS TO PROCESS 1980 OLYMPIC RESULTS

The results of more than 280 events at the 1980 Olympics will be handled by two ICL 2904 computers, soon to be installed in Moscow.

These computers will be used to store the results of various Olympic events, provide editing facilities, and distribute information on these events to five international news services — Associated Press, United Press International, Reuters, Agence France Press, and TASS, the Soviet news agency.

The two computers, valued at $1.74 million, are scheduled to be installed in Moscow in late April. ICL is providing a complete system design service to the 1980 Olympic Committee as well as technical advice and support throughout the implementation period and during the games.

A sub-set of the Olympic results system will be tested during the Spartakiada Games which take place at the Olympic Stadium in Moscow in July this year. Spartakiada, an athletic event for Eastern European countries, is expected to attract a great deal of European press interest because for the first time Western athletes will be invited to compete.

The configuration of the two ICL 2904's includes 96K words of store on each 2904, four EDS 60s, four magnetic tapes, two card readers, two line printers, two paper tape reader/punches, a 7502 terminal processor, and two modular protocol converters each with two asynchronous multiline communications controllers. All peripherals are switchable between the two 2904's, one of which provides a fail-safe facility.

COMPUTERLAND TO SELL SOL COMPUTER LINE

Computerland stores throughout the United States will sell and service the Processor Technology line of Sol microcomputers, according to a joint announcement by Robert Marsh, president of Processor Technology, and Richard Graham, manager of marketing for Computerland. The Computerland franchises now include seventy stores throughout the United States.

CALDISK PLANS ANNOUNCED

Billings Computer Corporation has announced plans for the floppy disk drive company which it acquired from CalComp (California Computer Products) earlier this year.

According to company president, Roger E. Billings, the operation has been named CALDISK. "The major thrust of the division will be to continue to supply floppy disk drives to the OEM market," he said. "In fact," he emphasized, "CALDISK will remain in California employing a majority of the same people who worked for CalComp before the acquisition."

Billings also said plans are underway to double production capacity within the next six months. The operation is presently shipping approximately 1600 single-sided, 8-inch drives per month.

INDIANA CLUB FORMS

A new computer club for users of all types of home computers has been formed in Evansville, Indiana. The group meets on the second Wednesday of each month at 7:30, usually at the Blind Association Building. Anyone interested in computers is asked to contact Robert Heerdink, National Sharedata Corp., P.O. Box 3895, Evansville, IN 47737, (812) 426-2725.

NEW ENERGY-SAVER

An energy-saving solid state clock thermostat with digital readout of time and temperature has been introduced by RapidCircuit of Brooklyn, New York.

The RC-3000 is completely electronic and requires little or no rewiring when it is used to replace a standard thermostat.

It turns the heat down at night and back up at a preset time, allowing from a 9% to 30% energy savings. The time and temperature are alternately displayed by LEDs. The temperature can be displayed either in Farenheit or Celsius.

For details write to the RapidCircuit Corp., 5721 18th Ave., Brooklyn, NY 11204. Collect phone calls are accepted at (212) 837-2424. Ask for Edward Carr.

FEELY ELECTED TO NATIONAL ACADEMY OF ENGINEERING

American National Standards Institute President Frank J. Feely, Jr., has been elected to the National Academy of Engineers, the most distinguished honor that can be conferred on an engineer.

The Academy recognizes those who have made important contributions to engineering theory and practice or who have advanced technology in new and developing fields.

Mr. Feely has been cited for his "Leadership in modernizing and strengthening national engineering standards, and in developing and applying petroleum technology." Feely is vice-president and director of Exxon Research and Engineering Company.

NEW BOOK OF "LAWS"

John Peers, Chairman of Logical Machine Corporation, has compiled *1001 Logical Laws, Accurate Axioms, Profound Principles, Trusty Truisms, Homey Homilies, Colorful Corollaries, Quotable Quotes, Rambunctious Ruminations for All Walks of Life.* John hopes that his efforts, besides providing pure entertainment, will enable his cohorts in the industry to see what life is really like "out there" where computers are daily office companions.

The laws were collected in response to an advertisement run by Logical Machine soliciting comments on life in general and life with business computers. Published by Doubleday the 189-page book costs $7.95.

FROM THE FOUNTAINHEAD

By Adam Osborne

Last year at the Second West Coast Computer Fair, I presented my first "White Elephant Award" for achievement in the microcomputer industry. I made the second annual award at the Fourth West Coast Computer Fair in May.

The name "White Elephant Award" is whimsical, but not its intent; I wish to honor outstanding achievement. But in an industry where "Kentucky Fried Computers" can grow into a company the size of North Star, where smaller is better, and where amateurs working in garages frequently stand a better chance at success than established corporations, it is only appropriate that my award also have a name with reverse connotations. My intent is as serious as the name is frivolous; my selections for the award are intended as the highest compliment that I can pay to a person or a product.

The White Elephant Award consists of a printed circuit card on which is mounted the "chip" of the year; in this instance the year will be calendar 1978, and any chip that first became available in 1978 is eligible for the award.

Last year, I presented the first White Elephant Award to Gary Kildall of Digital Research for the development CP/M, the defacto standard operating system of the microcomputer industry. In 1977 usable software of any type was the critical problem facing the microcomputer industry and Gary Kildall did more to solve the problem than anyone else.

But in 1978 we have encountered a new problem: the economic viability of companies which have grown from garage operations to multi-million dollar undertakings without simultaneously advancing their financial stability or management ability. Entering 1979, the microcomputer industry faces problems that are far more critical than most industry leaders are willing to face up to. Within the next 12 months, we can expect to see a large number of hardware manufacturers either go bankrupt or withdraw into a mode of restricted operations in consequence of cash flow shortages and mismanagement.

Tandy, Commodore and Heath are major entries today. Tomorrow we can expect Texas Instruments, Atari and others to make their presence felt. These established companies have been and will continue to be successful for two reasons: their own best efforts and the negligence fo the pioneers who preceded them. Established companies entering this market have the management, economic resources, and experience to capitalize on the scenario as they encounter it.

Unfortunately, they have received little or no resistance from the microcomputer manufacturers who pioneered this market, since these pioneers were unable to see the world change around them. Gone are the days when anything will sell, simply in consequence of being advertised. Gone are the days when for every seller there were a hundred buyers. Today, we are faced with an adequate supply of hardware manufacturers and a more sophisticated customer base which understands the difference between good and bad and has the options to exercise this understanding.

In the summer of 1978 almost any viable microcomputer hardware manufacturer could have obtained sound financing and seasoned management. Few of them did because they were too conceited or too ignorant to understand the true circumstances in which they were operating. Today, a few of the better microcomputer hardware manufacturers can still obtain financing — if they are willing to accept sound management with the package. But time is running out, and with it, an era of the microcomputer industry is passing.

But there are notable exceptions; and it is to these exceptions that I turn to identify the individual who we should all honor for making the most significant contribution to microcomputing in 1978.

Apple Computer Corporation is the company that has done the most outstanding job of bringing professionalism, sound management, and a first class product to the microcomputer marketplace. Mike Markkula, the president of Apple Computer Corporation, more than any individual is responsible for this achievement.

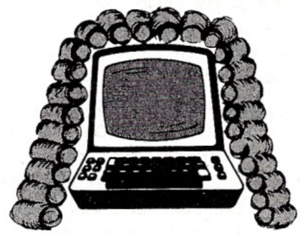
I therefore select Mike Markkula as the individual who has contributed the most to microcomputing in 1978. Steve Jobs and Steve Wosniak designed and built the first Apple, but it was Mike Markkula who took an operation which could have withered into just another case of mismanagement, and turned it into the first class company that Apple has become today.

In late 1976, Mike Markkula, then working for Intel, cashed in his Intel stock options worth approximately $250,000; he took himself and his money into Apple. Since that time, Mike has brought in professional management, sound business planning, and more equity financing as and when needed. The entire microcomputer industry would have been better off if more people had had Mike's foresight and understanding.

Harry Garland at Cromemco deserves Honorable Mention for his achievements. Cromemco is a low-key company, perhaps reflecting Harry Garland's low-key personality. But like Harry, Cromemco is technically superb and fiscally sound. When the dust of the coming 12 months settles, I expect Cromemco will be doing as well as ever.

I choose Intel's 8086 as the Chip of the Year for 1978. As the first of the new 16-bit microprocessors, this selection was not hard to make. In fact, Intel consolidated its position as leader of the microcomputer industry during 1978. Certainly Intel deserves to be recognized as having given us the outstanding product of the year: the 8086, delivered on time.□

JURISPRUDENT COMPUTERIST



By Leonard Tachner
Attorney at Law

THE PATENTABILITY OF COMPUTER SOFTWARE RELATED INVENTIONS — THE SUPREME COURT'S LATEST DECISION

This article updates a June, 1978 piece by Mr. Tachner which discussed legal protection of computer software. Since that time the Supreme Court has made a decision, explained here, which changes the status of software related developments.

A question of potentially great importance to the computer industry has again come to the forefront with the U.S. Supreme Court's latest patent decision in Parker versus Flook. For many patent attorneys, Justice Stevens' majority opinion of the six-to-three divided court creates more questions than it settles regarding the question of whether or not inventions relating to computer programs are protectable by patent.

This highly controversial and complex question became a substantial concern to the computer industry seven years ago when the U.S. Court of Customs and Patent Appeals (C.C.P.A.) decided In re Benson, a patent case in which the invention related to a method for converting one format of computer data into another format of computer data. The court found the Benson invention to be statutory subject matter even though the proposed patent claims included little or no physical apaparatus associated with the method steps recited in the patent application. The Patent Office sought and obtained review by the U.S. Supreme Court.

In November 1972, the Supreme Court reversed the decision of the C.C.P.A. holding that the claims to a method for converting one computer format to another were non-statutory subject matter. Unfortunately, the Supreme Court's decision in the Benson case was ambiguous, as evidenced by the many conflicting interpretations of that decision. The major computer hardware manufacturers and their representative associations, who do not want patents granted on programs for their machines, interpret the Benson decision broadly as a proscription on patents for software and software related inventions. Software development firms, on the other hand, advocate patent protection for their inventions and interpret the Benson decision narrowly as applying to only the specific facts of that case.

In the computer-program-related patent cases heard by the U.S. Court of Customs and Patent Appeals after the Benson decision, it has been evident that the judges of the C.C.P.A. are also split in their interpretations of the Supreme Court decision in the Benson case. The majority of C.C.P.A. judges has apparently sided with the software people by interpreting the fundamental rationale of Benson to be merely that a method encompassing all practical use of a mathematical formula and the involved algorithm constitutes non-statutory subject matter.

In the series of software related cases decided after the Supreme Court Benson decision, the C.C.P.A. has apparently begun to set guidelines for software related inventions. In one such case, In re Flook, the invention related to a process for controlling at least one parameter of a catalytic hydrocarbon conversion process in which an alarm value is periodically adjusted as a function of the history of the actual value of the parameter and the adjustment is accomplished by a computer in accordance with a mathematical control equation. The main claim of the patent application, which defines that the inventor believes to be patentable over prior art, recited a mathematical equation.

Despite this, the C.C.P.A. found the claim to be patentable subject matter because it included post-equation solution activity in which the solution of the equation was applied to a control system.

Thus, it appeared that at least one of the guidelines the C.C.P.A. was attempting to establish (based upon its interpretation of the Supreme Court's Benson decision) was that

if an invention relates to a process involving a computer program, and is defined as including a data processing system in combination with an external system under the control of a computer, it is acceptable to include the algorithm or mathematical equation in the definition of the invention as long as there is some non-trivial post-equation solution activity and the claim or definition of the invention does not end merely with the solution of the equation itself.

In the petition requesting review by the U.S. Supreme Court, then acting Commissioner of Patents and Trademarks, Lutrelle F. Parker, said that the decision of the C.C.P.A. regarding the Flook patent would have a debilitating effect on the rapidly expanding computer software industry and would require the Patent Office to process thousands of additional patent applications.

This argument seems to have struck a nerve. The U.S. Supreme Court granted certiorari; in recent years, an extremely rare event in patent related cases.

The Patent Office position, apparently undisputed by the patent applicant, was that the only difference between known methods of changing alarm limits and the method described in the application was in the mathematical equation or formula, which allowed an operator to calculate an updated alarm limit once he knew the original alarm base and other parameters regarding operating conditions of the catalytic conversion process. The court agreed and reasoned that the equation must be treated as though it were a familiar part of the prior art, whether it was in fact known or unknown at the time of the invention. The court ruled that the Flook claim was non-patentable subject matter and reversed the decision of the C.C.P.A. In a footnote to the decision, Justice Stevens summarizes the holding of the court, stating:

> "Very simply, our holding today is that a claim for an improved method of calculation, even when tied to a specific end use, is unpatentable subject matter under Section 101."

Justice Stewart, joined by the Chief Justice and Justice Rehnquist in his dissenting opinion, criticized the majority for misinterpreting the provision of the Patent Statute which relates to the question of what is patentable subject matter by holding, in effect, that a claimed process loses its status of patentable subject matter simply because one step in the process is not patentable if considered in isolation. In effect, the dissenting opinion accuses the majority of confusing the question of novelty and inventiveness with the question of what subject matter is protectable by patents in the first place.

Thus the court appears to be saying that, at least as to patent protection for computer programs. . . Congress must act. . .by specifying. . .patentable subject matter. . .

In any case, this latest decision of the U.S. Supreme Court regarding the protection of computer programs and computer program related inventions does not finally settle the question, except perhaps as to computer programs per se. The majority in Flook stated:

> "Neither the dearth of precedent, nor this decision, should therefore be interpreted as reflecting a judgement that patent protection of certain novel and useful computer programs will not promote the progress of science and useful arts, or that such protection is undesirable as a matter of policy. Difficult questions of policy concerning the kinds of programs that may be appropriate for patent protection and the form and duration of such protection can be answered by Congress on the basis of current empirical data not equally available to this tribunal."

Thus the court appears to be saying that, at least as to patent protection for computer programs per se, Congress must act first by specifying that this is patentable subject matter, before the Supreme Court will find it to be so under the current patent statute.

However, the question of patentability of computer program related inventions, that is, those inventions that are not defined as mere programs alone, but systems that utilize programs in conjunction with hardware, is still one of great importance that remains unsettled.

For those who seek a definitive answer from the various court decisions on this question, interpret this latest decision as one more bit of information which helps set guidelines patent practitioners may use to draft patent claims to inventions that include or relate to computer programs. This one additional bit of information seems to be that a process which appears advantageous only because it includes a new method for calculating that permits the application of a high speed computer to accomplish that calculation, will not be patentable.

In other words, for it to be patentable, there must be something more novel in an inventive process than merely a previously unknown equation especially suitable for solution by a computer.☐

The material presented in this column is intended for the reader's generaly information. The author requests that the reader consult professional advisors prior to applying this material to his or her specific situation. Anyone seeking further information may contact the author at the Law Firm of Fischer and Tachner, 2192 Dupont Drive, Suite 210, Irvine, CA 92715.

CIRCLE INQUIRY NO. 14

MICRO MEDICINE

By Wm. V. Weiss, M.D.
Professional Engineer

I was recently discussing back problems with an associate who does occupational medicine for a large brewery where back problems present 30% of all employee injuries and are most costly in terms of lost productivity and suffering. Since we had begun to consider this problem at the same time, I suggested he have his company commission a review of the subject.

At the same time we initiated the study of back problems, I happened to catch an advertisement on television showing medical research work being done on back problems in Chicago. The TV ad was most intriguing, showing a conceptualized line drawing via computer graphics simulation of a human spine during various different maneuvers. The ad left the impression that someone was really on the track to quantifying and characterizing the puzzling problems of back disease. I began to feel somewhat optimistic that light lay ahead at the end of the "back problem" tunnel.

After considerable effort I finally identified the principal investigator, a Professor Albert B. Schultz (Professor of Mechanical Engineering) of the University of Illinois at Chicago Circle. I wrote him inquiring into the nature of his investigations and the type of system he was using for his simulations.

Before returning to Professor Schultz let's digress a moment and discuss the back and spine. Even presently this is a poorly understood structure. The development of the vertebrates spans some 500 million years, and that of modern man about 40,000 generations. The evolution from four-footed animal to upright man resulted in the original "bridge structure" rotating upwards into a "column structure."

Until very recently (1960's) the only knowledge medical people had of the spine was the qualitative anatomical descriptions of the anatomists. Most of this material was not truly representative of the living organism as it had developed from cadaver studies. Even our anatomical studies are incomplete due to the incredible complexity of the various spinal elements strung together in a web not unlike the rigging of a "tall ship." The most elaborate "biodynamic models" to date are only simple approximations of the various static and dynamic forces at play. Instrumentation of the spinal vertebral elements (the bony sections) has been extremely difficult in the living body and thus the "yield strenghts" of most tissues and bones comes from cadaver studies, and are approximate at best.

In humans, one of the great puzzles has been that the mechanism of injury (disks primarily) is not always related to trauma, straining, or other obvious "unusual" force, as 35-40% of all disk problems occur in sedentary individuals without *obvious* cause.

Returning to my correspondence, Professor Schultz was most gracious in his reply. He informed me that his aspirations ran parallel to mine, with prevention and prognostication high on the list of practical applications, but that there were still enormous gaps in the basic theory required to understand the physical limiations of the various tissues and that many years of empirical work was yet required to determine the tensile, rotational and shearing forces that the spine can sustain in normal life. Then, of course, further study would be required of man in the infinite variety of postures "working man" finds himself before the data base would be adequate for valid mathematical formulations.

His current work requires the vast computational power of the largest mainframe systems due to the number of equations run simultaneously. If one starts to conceptualize a model consisting of the forces of a moving human body with analogs of mass, springs, damping in three dimensions, it becomes obvious it cannot be run easily on a home computer.

In order to build a mathematical model of the spine, extensive empirical measurements are necessary to develop some quantitive guidelines to the relationships between various postures and their respective spinal effects. Some of the most time-consuming experimental work (with cadavers, primates, and human volunteers) requires the collection and analysis of stresses on many locations of the spine. Remembering that our spinal column has 24 mobile bones (vertebrae) with seven in the cervical or neck area, 12 in the thoracic or chest region, and five in the lumbar or low-back region. The normal spine is somewhat "S" shaped with the degree of curvature (in the anterior-posterior plane) a function of fitness and genetic endowment. As one can imagine, almost every posture and position of arms (and load) creates a panorama of changing stresses at various levels.

To confuse the issue even more is the variability of response of human tissues to seemingly similar conditions. A perfect example is the "slipped disk." This is a misnomer as the disk rarely moves, but rather, some of the central gel material extrudes outwards through a weakened portion of its containing wall, causing local pain and "referred pain" if it presses on one of the spinal nerve "roots." Why is it that of two different people of seemingly similar physical characteristics (height, weight, build, sex, age, etc.) one can suffer a "herniated" or "ruptured" disk lifting a newspaper and the other does not?

Although biological characteristics are more or less similar, there is no comparison to inanimate materials like steel where the tensile strength is a highly predictable characteristic for similar alloys. Thus, it appears that genetically some people have "tougher disks" than others, and this must explain the wide variation in tolerance to different work conditions. Obviously, when one considers the variety of positions a worker, or even a gardener, subjects himself to in the course of an activity it is almost impossible to compare two victims of back strain. Thus, prognostication is extremely difficult.

It suffices to comment that it is good that there are investigators patient enough to slowly pick away at all of the unknowns in this field. Hopefully a "sample" of the infinite numbers of stresses at play will allow accurate predictive models in a reasonable time frame. I can plead guilty to perhaps expecting too much as I briefly saw these marvelous spinal segment graphics simulations flexing and bending in response to electronic forces on the screen last year.

My second mistake (wishful thinking) was to assume my associate would turn up some good news in his literature review, in spite of my disappointment at the rate of "model" or mechanics developments. At present it seems there is no reliable method of predicting who is most likely to suffer back problems (work related or otherwise). Worse still, there is no way of preventing such problems even in sedentary workers, and "back training clinics," and "lifting instruction" have had very little impact on the rate of new injuries. X-ray of potential employees, once thought to be useful in weeding out some potential back injuries, is now considered a waste of money as a screening technique. At this point, we can only promote general fitness, try and tailor the likely stresses a person might encounter to the appropriate body-build and capacity, and gradually "engineer" the work environment such that peak spinal stresses will be avoided.

To end on an optimistic note, I hope that in the near future research will have had advanced to the point where a short "Back Program" for home comptuers can be produced. Starting with some input data such as age, sex, height, weight, girth at chest, hip and waist, one might carry on a "dialogue" which will ask what type of physical tasks you are anticipating and, based on certain additional data (load to be lifted, various positions, etc.), will tell the interrogator the limits his personal capacity imposes if he wishes to avoid back injuries. Later versions would hopefully include some graphics for greater clarity.☐

Dr. Weiss can be contacted at Biolithics, 600 Sherbourne Street, Suite 803, Toronto, Ontario M4X 1W4.

CALL FOR ARTICLES

INTERFACE AGE is seeking articles on hardware and software, particularly concerning Apple, TRS-80, Atari and Heath computer systems. Other areas of interest include medical and educational applications, special functions of microprocessors, telecommunications and video graphics, both on screen and hardcopy.

The payment rate for articles ranges from $20 to $50 per published page. Pieces describing company projects or products will carry the company byline, but no payment is offered. Submittals should include an abstract, outline and stamped return envelope.

Manuscripts should be typed, double spaced with 1-inch margins. Minimum length is four pages, unless programs are included. Photos should be numbered and have a brief description attached. Tables, listings, etc. should be on separate pages. Computer listings should be printed using a new ribbon to assure good reproduction. Authors are requested to submit a statement of their background and expertise.

The publisher assumes no responsibility for artwork, photos or manuscripts. No acknowledgement is made unless accompanied with a stamped return envelope.

For article submission or more information, contact Carl Warren, Editor-in-Chief, INTERFACE AGE Magazine, 16704 Marquardt Ave., Cerritos, CA 90701. Please do not phone.

THE MICRO-MATHEMATICIAN

By Dr. Alfred Adler

PROGRAM CONVERT

In the March issue this column presented a HEX to decimal conversion chart which greatly simplifies many programming tasks. However, since such conversions may be needed during program execution, this month we are presenting a short routine (written in PolyMorphic BASIC, Version A00), that will convert binary, octal, decimal, or HEX to binary, octal, decimal, or HEX.

There are many ways in which such a general conversion program can be written. If we had attempted to make direct conversion from any of the four systems to any other of the four, 12 separate conversion algorithms would have been needed. If on the other hand we take all conversions through decimal, that is, whichever number system is used for the input, it is first converted to decimal (unless it is already decimal) and then converted from decimal to whatever output is required (unless decimal output is required), then only six separate algorithms are needed. This is the approach taken for Program CONVERT.

The algorithms needed to make these various conversions follow directly from a consideration of the structure of the various number systems. Starting with the binary system, note that the least significant bit tells us how many ones, the next significant bit tells us how many twos, the next how many fours, then how many eights, etc., doubling each time we move to the next most significant bit. In order to convert this binary representation to decimal we must therefore multiply each bit by two to the z power, where z equals one less than the column number, counting left from the least significant bit. Thus the 1 in binary 1000 equals 1 times 2^3, since the 1 is in the fourth column from the left and three is one less than four. Each bit is converted in turn and the results added. This algorithm appears in statement numbers 210 to 235 of the program.

To convert octal to decimal, note that again the least significant digit tells us how many ones, but now the next significant digit tells us how many eights, and the next how many sixty-fours, and so forth. In order to convert this octal representation to decimal we must multiply each digit by eight to the z power, where z again equals one less than the column number, counting left from the least significant digit. Thus the 4 in octal 3421 equals 4 times 8^2, since the 4 is in column three and two is one less than three. This algorithm appears in statement numbers 305 to 330 of the program.

To convert HEX to decimal, note that now each successively more significant digit tells us how many ones, sixteens, two-hundred-and-fifty-sixes, etc. To convert the HEX representation to decimal we multiply each digit by sixteen to the z power, where z is the same as before. An additional problem arises, however, Suppose we want to convert 5FC3 to decimal. The F in HEX cannot be converted to decimal until we change the F into a number. We all know that it equals 15 decimal, but the computer doesn't know it.

A quick and easy way to take care of this problem is through the use of the ASC function in BASIC, as is done in statement numbers 425 and 430 of the program. If the ASC function is not available in your particular BASIC, statement numbers 425 and 430 must be replaced by the less elegant but equally serviceable statement numbers 425 to 455 shown in Figure 2. The complete algorithm for converting HEX to decimal appears in statement numbers 410 to 480 of the program.

Now that our binary, octal, or HEX number is nearly converted to decimal, we must be concerned with converting the decimal number into binary, octal, or HEX, as required. Once again the algorithms follow directly from a consideration of the structure of the various number systems.

Considering the first conversion of decimal to binary, we will assume that the binary number will be no more than sixteen bits long, since most of us are limited to 64K address lines. We proceed from left to right, that is, from most significant bit to least significant bit, essentially reversing the procedure we followed above in converting to decimal. Whereas before we multiplied each bit by two to the z power, we will not divide each bit by two to the z power, where z is still defined as before.

Since we are only interested in whole values of the division, we INTEGER the quotient. We then decrease the decimal number by the product of the integered division and the current value of z. For example, to convert 24617 decimal to binary we proceed as follows. Since we are dealing with a sixteen bit binary number, the most significant bit tells us how many 2^{15}s we have. We therefore divide 24617 decimal by 2^{15} and INTEGER. The result is zero. This says that there are no 2^{15}s in decimal 24617. Therefore we place a zero in the leftmost column of the sixteen digit binary number. We then divide by 2^{14} and INTEGER. The result is a one. This tells us that there is one 2^{14} in decimal 24617, therefore the second digit from the left in the binary number is a one.

Now we must subtract 2^{14}, which equals 16384 decimal, from 24617. This leaves decimal 8233 which we now divide by 2^{13} and INTEGER. The result is a one. We place a one in the third column from the left of the binary number and subtract 2^{13} from 8233. The process is continued until either the remainder is reduced to 0 decimal or until we have divided by 2^0 (which equals one). This latter operation will coincide with the right hand most column of the binary number; that is, the least significant bit. This algorithm appears in statement numbers 60 to 80 of the program.

Conversion from decimal to octal is done in a similar manner with two exceptions. First, the octal number need only be six digits long to correspond to a sixteen bit address. Second, since we are now using a base eight number instead of a base two number, we use eight to the z power instead of two. For example, to convert 24617 decimal to octal we first divide 24617 by 8^5 and INTEGER. The result is zero, telling us that there are no 8^5s in decimal 24617. We place a zero in the leftmost column of the six digit octal number. We now divide by 8^4 and INTEGER. The result is a six, telling us that there are six 8^4s in decimal 24617. We place a six in the second from the left column of the octal number, subtract 6 times 8^4 (24576) from 24617 and proceed. The algorithm for conversion of decimal to octal appears in statement numbers 100 to 120 of the program.

Conversion from decimal to HEX is again similar, with three differences now. First the HEX number need only be four digits long. Second, we must use sixteen to the z power, since we are dealing with a base sixteen number. And third, any quotient exceeding nine must be converted to a letter between A and F, in accord with the hexadecimal conventions.

For example, to convert 24570 to HEX we first divide by 16^3 and INTEGER. The result is a five, which becomes the first digit of our four digit HEX number. Subtracting 5 times 16^3 from 24570 we are left with 4090 decimal. We divide this by 16^2 and INTEGER, obtaining 15, which is really the second digit of our HEX number. Since it would be quite confusing to try to use a two digit symbol to represent a single digit, convention requires that we express the 15 as an F. The algorithm for making such changes appears in the program in statement numbers 165 to 173. The complete algorithm for converting decimal to HEX appears in statement numbers 150 to 180.

In use, the operator will be asked to state the input base, the output base, and the input number. The program will respond with the input number converted to the new base, and will recycle awaiting a new request. Sample runs are presented following the listing.□

Alfred Adler can be contacted at 10360 Flintlock Trail, Tucson, AZ 85715.

Program Listing Follows

Basic Time INTRODUCES THE $15,995* 10MB HARD DISK COMPUTER

*Quantity 1 Price

Complete System Includes:

★ 10 MILLION CHARACTER HARD DISK STORAGE
★ DISK OPERATING SYSTEM & EXTENDED BASIC
★ POWERFUL 16-BIT CENTRAL PROCESSING UNIT
★ VIDEO DISPLAY TERMINAL
★ 150 CPS LINE PRINTER AND PRINTER STAND
★ DESK ENCLOSURE

basic time

1215 E. El Segundo Boulevard
El Segundo, California 90245
213 / 322-4435

PROGRAM LISTING

```
>>LIST
1 REM<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<>>>>>>>>>>>>>>>>>>>>>>
2 REM
3 REM<<<<<<<<  P  R  O  G  R  A  M    C  O  N  V  E  R  T
4 REM
5 REM<<<<<<<<<<<<<<<<  VERSION  1.1  <<<>>>  MAY  1978  >>>>>
6 REM
7 REM<<<<<<<<<<<<  WRITTEN  BY  -  ALFRED  A.  ADLER  PHD  >
8 REM
9 DIM B(16),N(17),B$(16)
10 INPUT "STATE INPUT BASE :B,O,D,H : ",I$
15 INPUT "STATE OUTPUT BASE :B,O,D,H :",F$
20 IF I$="B" THEN 200
25 IF I$="O" THEN 300
30 IF I$="H" THEN 400
35 REM I$="D"
40 INPUT "STATE DECIMAL NUMBER :",N(1)
45 IF F$="H" THEN 150
50 IF F$="O" THEN 100
53 IF F$="D" THEN !N(1)\!\GOTO 10
55 REM F$="B"
60 FOR K=1 TO 16
65 M=2^(16-K)
70 B(K)=INT(N(K)/M)\N(K+1)=N(K)-B(K)*M
75 !B(K),
80 NEXT
85 !\!\GOTO 10
95 REM F$="O"
100 FOR K=1 TO 6
105 M=8^(6-K)
110 O(K)=INT(N(K)/M)\N(K+1)=N(K)-O(K)*M
115 !O(K),
120 NEXT
125 !\!\GOTO10
145 REM F$="H"
150 FOR K=1 TO 4
155 M=16^(4-K)
160 H(K)=INT(N(K)/M)\N(K+1)=N(K)-H(K)*M
165 Q$="ABCDEF"
170 A=H(K)-9
173 IF A=>1 THEN !" ",Q$(A,A),\GOTO 180
175 !H(K),
180 NEXT
190 !\!\GOTO 10
200 REM I$="B"
205 INPUT "STATE BINARY NUMBER :",B$
210 D=0\J=LEN(B$)
215 FOR K=1 TO J
220 M=2^(J-K)
225 IF B$(K,K)="0" THEN 235
230 D=D+M
235 NEXT
240 N(1)=D
243 IF F$="B" THEN 60
245 IF F$="O" THEN 100
250 IF F$="H" THEN 150
255 !D\!\GOTO 10
295 REM O$="O"
300 INPUT "STATE OCTAL NUMBER :",O$
305 D=0\J=LEN(O$)
310 FOR K=1 TO J
315 M=8^(J-K)
320 O(K)=VAL(O$(K,K))
325 D=D+O(K)*M
330 NEXT
335 GOTO 240
400 REM I$="H"
405 INPUT "STATE HEX NUMBER :",H$
410 D=0\J=LEN(H$)
415 FOR K=1 TO J
420 M=16^(J-K)
425 H(K)=ASC(H$(K,K))-55
430 IF H(K)=<9 THEN H(K)=H(K)+7
475 D=D+H(K)*M
480 NEXT
485 GOTO 240
>>RUN

STATE INPUT BASE :B,O,D,H : D
STATE OUTPUT BASE :B,O,D,H :H
STATE DECIMAL NUMBER :65535
 F F F F

STATE INPUT BASE :B,O,D,H : H
STATE OUTPUT BASE :B,O,D,H :D
STATE HEX NUMBER :FFFF
 65535

STATE INPUT BASE :B,O,D,H : D
STATE OUTPUT BASE :B,O,D,H :B
STATE DECIMAL NUMBER :65535
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

STATE INPUT BASE :B,O,D,H : B
STATE OUTPUT BASE :B,O,D,H :H
STATE BINARY NUMBER :0101010101010101
 5 5 5 5
```

Branched to Page 40

CIRCLE INQUIRY NO. 95

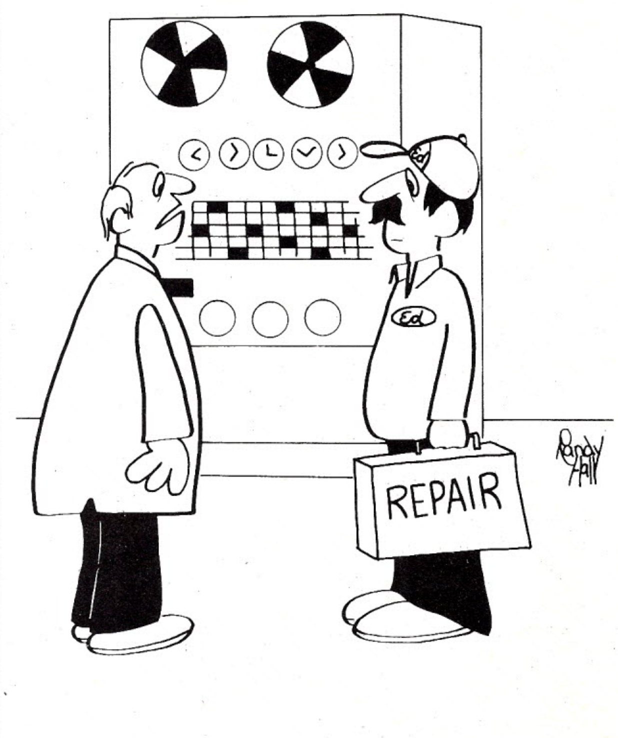
Vectored from Page 38

```
STATE INPUT BASE :B,O,D,H : H
STATE OUTPUT BASE :B,O,D,H :B
STATE HEX NUMBER :AAAA
 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0

STATE INPUT BASE :B,O,D,H : B
STATE OUTPUT BASE :B,O,D,H :O
STATE BINARY NUMBER :1010101010101010
 1 2 5 2 5 2

STATE INPUT BASE :B,O,D,H : H
STATE OUTPUT BASE :B,O,D,H :O
STATE HEX NUMBER :FFFF
 1 7 7 7 7 7

STATE INPUT BASE :B,O,D,H : O
STATE OUTPUT BASE :B,O,D,H :D
STATE OCTAL NUMBER :76543
 32099

STATE INPUT BASE :B,O,D,H :
Interrupted in line 10
>>
```

LISTING 2

Substitution for statement numbers 425 and 430, for Basic interpreters that do not contain the ASC function.

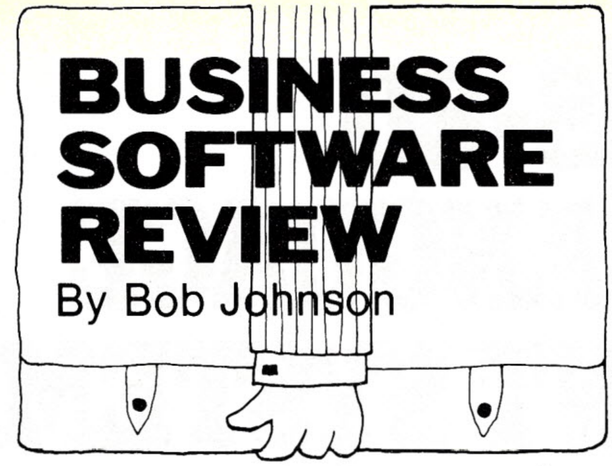
```
425 IF H$(K,K)="A" THEN H(K)=10\GOTO 475
430 IF H$(K,K)="B" THEN H(K)=11\GOTO 475
435 IF H$(K,K)="C" THEN H(K)=12\GOTO 475
440 IF H$(K,K)="D" THEN H(K)=13\GOTO 475
445 IF H$(K,K)="E" THEN H(K)=14\GOTO 475
450 IF H$(K,K)="F" THEN H(K)=15\GOTO 475
455 H(K)=VAL(H$(K,K))
```



"It looks like a computer, but it's acting like an abacus."

BUSINESS SOFTWARE REVIEW

By Bob Johnson



New and prospective users of small business computers frequently become confused with some of the concepts and terminology thrown at them while shopping for a system. Some of these concepts are extremely important to the person shopping for a business system.

In my experience, one of the hardest things for a neophyte to comprehend is the difference between operating systems, languages, monitors, and application programs. These four terms are bandied about by computer dealers as if the prospective customer had used them all his life. Even more confusing to some buyers is the conceptual separation between the hardware (the actual computer system, terminals, printers, etc.) and software (which includes all programs to be run on the machine).

More confusing still is the fact that small systems are usually a collection of different products which have been combined together to form a complete system. It is not unusual to see a system consisting of, say, a Cromemco C-3 computer, a Hazeltine 1500 printer, a Texas Instruments 810 printer, a D.C. Hayes & Associates Modem, CP/M, and two or three other suppliers' software. (This particular combination is used strictly as illustration.)

So, if you newcomers to the wonderful world of computer jargon will hang on, I'll try to quickly define some of these terms. First of all, hardware refers to anything which could be considered a machine. This will include the actual computer (or mainframe), the terminals, or the printer. Hardware in itself can do no useful work. All hardware is controlled by programs, and these programs can be called by many various names. By the way, just to confuse the issue, some dealers will call hardware the "system" (I confess — I've done it, too. . .), while others will refer to the entire combination of hardware and software as the "system."

There are several types of "programs" which are used by your computer system. The first type of program, and the most basic, is usually called "firmware." Basically, firmware is a small program which resides in ROM (or Read-Only Memory). This ROM type of memory does not lose the program when the power is turned off. Therefore, it is ideal for telling the computer how to do some very simple tasks when the power is first turned on. Usually, these tasks will include giving you some sort of message on your terminal, doing some very basic machine language programming, and getting your disk drives started. This very simple program, or firmware, is called a "monitor" or "system monitor." It is necessary to let your computer know that it is a computer, and what to do first whenever the power is turned on.

The next level of program usually is loaded into the computer and run by the monitor. It is called an "operating system." The operating system handles all the basic details of your computer's operation. It has all the routines for storing information on the diskette drives and loading information from them into memory. It has the routines for driving your printer. It also allows you to manage several other types of programs and data. Sometimes the operating system is called an "executive" or on larger computers, a "job-control system." One of the popular operating systems for microcomputers is CP/M by Digital Research.

At the next level of programs, we find "languages." A language is a program which affords us humans some kind of common ground with the computer. Computers think in binary, or base two. The only thing that a computer can recognize is whether a particular spot in memory contains an "on" or "off" state. To a computer, the letter 'G' is composed of eight "switches," called "bits," and looks some-

THE TRS-80 ASSEMBLY LANGUAGE DEVELOPMENT SYSTEM.

(A STEAL AT TWICE THE PRICE)

A short time ago, Microsoft introduced TRS-80 FORTRAN— a complete ANSI-standard FORTRAN with macro assembler, linking loader, and text editor, all for only $350. **The response has been overwhelming.**

Many TRS-80 users even told us, "The assembly language development software alone is worth that price." We think they're right, of course, but we've made it an even better deal.

ANNOUNCING: THE TRS-80 ASSEMBLY LANGUAGE DEVELOPMENT SYSTEM FOR $175.

For half the price of the TRS-80 FORTRAN Package, you can buy the TRS-80 ASSEMBLY LANGUAGE DEVELOPMENT SYSTEM, including:

EDIT-80 A fast, random access text editor that's easy to use and loaded with features. Lets you insert, replace, print or delete lines; edit individual lines; renumber lines in a file; and find or substitute text.

MACRO-80 The best Z80 assembler anywhere. MACRO-80 supports a complete Intel-standard macro facility plus many other "big computer" assembler features: comment blocks, octal or hex listings, 8080 mode, titles and subtitles, variable input radix (base 2 to base 16), and a complete set of listing controls.

LINK-80 Loads your relocatable assembly language modules for execution and automatically resolves external references between modules.

CREF-80 Gives you a complete dictionary of program symbols, showing where each is defined and referenced.

The Microsoft TRS-80 FORTRAN Package is still available for $350. Or, for HALF PRICE, get the TRS-80 ASSEMBLY LANGUAGE DEVELOPMENT SYSTEM.

Either way, it's a steal.

TO: Microsoft, 10800 N.E. 8th, Suite 819, Bellevue, WA 98004

☐ Send me the works! TRS-80 FORTRAN Package for $350.

☐ Send me half the works! TRS-80 Assembly Language Development System for $175.

☐ Check enclosed ☐ Master Charge ☐ VISA

CARD NUMBER EXP. DATE

CARDHOLDER'S SIGNATURE

NAME

ADDRESS

CITY STATE ZIP

MICROSOFT

Dealer Inquiries Invited

thing like 01000111. Humans tend to go slightly batty when trying to think in this type of code for periods longer than ten or twenty or minutes. So languages were invented to sort of "bridge the gap." Languages range in complexity and structure all the way from "machine language" (in which you might specify the letter 'G' as 47H) through BASIC, in which you give the computer instructions in English-like commands such as <100 A=B+C : PRINT "THE ANSWER IS ",A>. Whenever most folks talk about programming a computer, they are talking about using a language in order to teach the computer how to perform some task. Some common languages are BASIC, COBOL, RPG/II, APL, SNOBOL, LISP, FORTRAN, PASCAL, and JOVIAL.

The highest level of programs are called "application programs." These are the ones that a businessman is really interested in. These are programs that will allow the computer to do tasks such as your inventory control or general ledger. If you are buying a computer for business, these type programs should be the deciding factor on which computer you buy.

The other term that sometimes confuses is "software." Software is any type of program other than firmware. Sometimes I think that half the effort in learning computers went into mastering the jargon.

THIS MONTH'S SOFTWARE — A.S.I. & MICROTEX

One of the reasons that I wanted to discuss terminology this month is that is directly relates to the software I am reviewing. Almost everyone has heard of the CP/M operating system, and it is pretty much a de-facto standard for microcomputer users. The accounting packages reviewed this month run under the OPUS language/operating system from Administrative Systems Inc. in Colorado. One is a complete Clinical Accounts Receivable and Billing package from A.S.I. The other is a General Contractor's accounting and job costing system from Microtex of Houston, Texas. Both of these packages are designed especially for a specialized market.

ADMINISTRATIVE SYSTEMS INC.

Administrative Systems started in 1973 with a mind to put together some business packages using the then-new microcomputers such as the Altair 8080. They were impressed with the capability of the machine and saw a great future in store for micros and business.

No one had released a good extended disk BASIC in 1973, so the people at A.S.I. decided to write a high-level language expressly designed for business applications programming and soon OPUS was born. It started off as a BASIC language, but soon outgrew many of the restrictions inherent in BASIC.

Over a period of several years they have improved and expanded it until OPUS has become an extremely powerful language, and one that a professional programmer can really appreciate. However, it is so flexible that a novice can get into trouble designing large packages with it. Consequently, most of the work currently being done with OPUS is being done by serious, professional programmers, and most of the packages available to date have been aimed toward a specialized market.

OPUS would be termed a "semi-compiler" language. In other words, after a line of code has been entered, it must be compacted and put into a format more suitable for the machine's operation. OPUS does not compile its code completely down to machine language such as COBOL or FORTRAN, so it and languages like it are called semi-compilers.

A couple of the nice features of OPUS are a simplified operating system called FORTE, which is designed to handle only the running of programs, and a complete time-sharing operating system called TEMPOS, which is upward-compatible from OPUS.

Versions of OPUS and FORTE are available for most diskette or cassette based systems. TEMPOS requires the use of some additional hardware, and I would suggest that a dealer set up a TEMPOS system. It can be a little too much for a new user.

A.S.I. Clinical Accounts Receivable and Billing

This package is designed to handle the billing and receivables for medical or dental clinics. It was originally designed and implemented on a DEC PDP-11 in 1974. Since that time, it has been converted to run on 8080 based microcomputers under A.S.I.'s OPUS. It has undergone several revisions and additions since its original conception.

The CAR/B system is a large, complex series of programs and currently is being offered only through authorized dealers. It will require that a dealer perform a great deal of software modification to

set it up for a specific clinic. Most of this modification is involved with setting up the proper files and formatting specialized reports, including up to five separate insurance forms.

Information kept on file by the system includes all of the following:

1. Doctors file — includes name, address, personal information.
2. Client file — name, address, activity history, financially responsible party, billing information, and receivables status.
3. Services file — contains detail of all services offered by the clinic and charges for said services.
4. Diagnostic file — contains detail of most diagnoses normally incurred.
5. Transaction file — contains daily records of activity, and is used to post to all other files.
6. Miscellaneous file — several miscellaneous files are kept containing information such as messages to be printed on statements, etc.

One of the strongest points of this system is the ability to not only control the billing of a small to medium sized clinic, but to provide reports of all billable activity within the clinic by service, doctor, or just about any other categorization desired.

The CAR/B system automatically generates the patients' monthly statements, insurance forms, and activity reports. Statements may also contain special messages, selected by payment type. Payment types include cash, insurance, and open accounts, and the user may define other payment types if desired.

Included with the CAR/B system is an inquiry system which allows the operator to access any information on file and define the format of specialized reports. A typical use of the inquiry function would be to obtain a list of all patients who have had lab work done in the last three months.

All in all, this system is difficult to describe in a vest-pocket review. If you are interested in this system, I would recommend that you call A.S.I. at (303) 755-9694 and ask for the name and number of the nearest dealer.

GENERAL CONTRACTOR'S ACCOUNTING PACKAGE

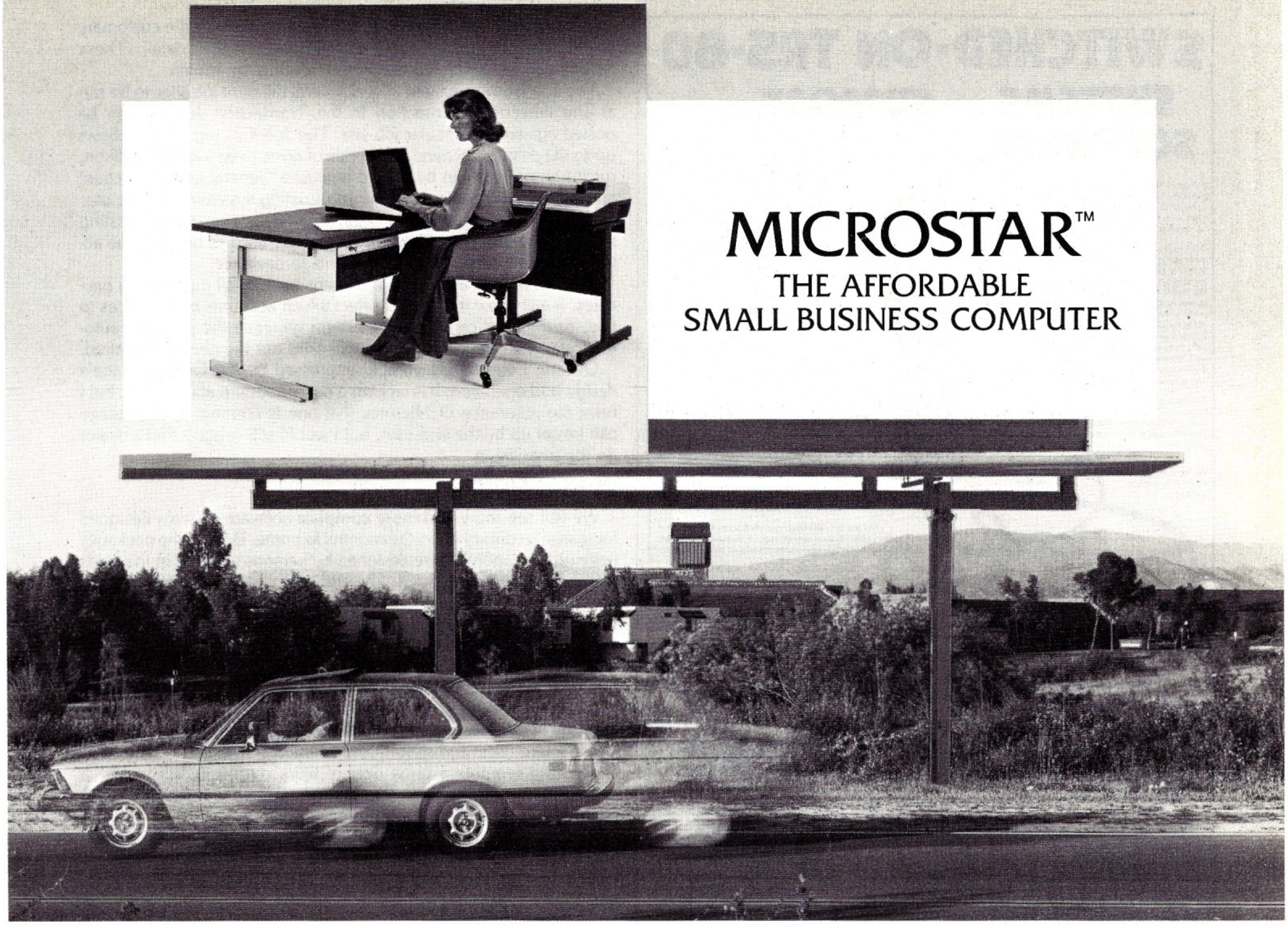
The Microtex General Contractor's Package designed by Gary Boucher was written especially for the contracting industry. The GCP is a complete accounting system consisting of a General Ledger, Accounts Payable, and Job Costing Journal. The entire system is menu-driven and all the inputs are error-checked.

The system is brought up under OPUS or FORTE by typing 'LOAD, "MAIN", RUN'. After that, all processing options are selected by choosing the number of that function from a menu of functions displayed on the CRT screen. I may be partial to this package, because Mr. Boucher's programming style is almost exactly like my own. He incorporates things I like, such as extensive screen formatting using direct cursor positioning, menu selection, and data-entry procedures designed for persons without computer experience.

The entire data entry process in any portion of the package is completely self-explanatory. And, if a mistake is made during entry, the user has two chances to correct it before it is processed. The first is the ability to back up to a previous entry is a mistake was made. Then, after all data has been input, the system prompts the user to check the information entered, and asks if any changes are needed. If changes are required, the user specifies which piece of data needs correcting, and the program goes to that portion of the screen to allow for corrections.

This may sound complicated on paper, but it produces two beautiful results. The first is that many people are initially overwhelmed by the speed and clarity in which data is displayed and input, and the second is the fact that an operator can enter data very quickly and with a good deal fewer mistakes than if the system did not do screen formatting.

All transactions are posted to the system through the Transaction Journal. The user defines whether the input is an Invoice, a Check, a Deposit, or a Journal entry. From this point, the system distributes the information to the appropriate journals automatically. Also, the Transaction Journal employs line-item entry, which I find very helpful. Line-item accounting refers to the process in which a transaction such as a check or invoice will be broken down into several different amounts and those portions posted against different accounts. The Transaction Journal allows up to 10 lines to be entered under each separate transaction.



TEST DRIVE ONE TODAY!

If you're in a big business, you know you can't operate without a computer. If you're in a small business, you think you can't afford one or that you can't afford to hire computer people to run it.

Voila! Enter MICROSTAR, the affordable small business computer. Put your own complete system to work in your office for under $10,000, or lease one for under $300 per month. Hire no computer experts – let your secretary learn to work the system in hours. Eliminate all that tedious, time-consuming manual accounting work. With MICROSTAR you can fully automate your accounts receivable, accounts payable, general ledger, payroll, inventory, and order entry. There's even a word processor to simplify your communications. Just think – no waiting for days, weeks, or months to check the color of your accountant's ink. You start guiding your P & L daily instead of monthly.

Of course, there are other microcomputers, but there is only one MICROSTAR. It's the computer with a single-board instead of the customary 6 or 8 printed circuit boards. And that gives you reliability and maintenance confidence not possible with other systems.

Ask us about such MICROSTAR features as: STARDOS, UPDATE, and CP/M compatibility. Let us explain the MICROSTAR multi-user capability that allows you to utilize simultaneously additional video display terminals and one line printer. Find out how easily our dealers can develop specific programs for your particular business.

So if you are a small businessman who never expected to computerize in 1979, call today for the name of your nearest MICROSTAR dealer. He's got a system just waiting for you to test drive.

Dealers: Join the "Micro Five Dealer Drive!"

MICRO V

MICRO V CORPORATION,
17777 S.E. Main Street, Irvine, California, 92714, Phone: (714) 957-1517, Telex: 678401 TABIRIN

CIRCLE INQUIRY NO. 42

The General Ledger system provides the user with the customary Profit/Loss, Balance Sheet, and General Ledger detail. These reports can be generated at any time during the month.

The Job Costing system allows several different job sites to be run at any time. Any transaction in the Transaction Journal can be posted against a particular job site. The Job Costing system allows up to 40 different general categories of costs to be kept. In addition, most transactions can be posted against a General Ledger account and a Job Costing account. The Job Costing system allows the user to enter initial cost projections for a particular site, and all costing reports show the projected amount (can be used as budget), the actual amount to date, and the percentage deviance.

The Accounts Payable system keeps track of all outstanding payables. It will allow the user to select which accounts and invoices to pay, or the user can have the system generate the payables automatically. The Payables system will print out the checks if desired.

Overall, I was very favorably impressed by the GCP package's design and operation. It is lacking a payroll system at this point, but I have the assurance of Microtex that one is coming. This package can be set up by the end user, but I would still suggest that a dealer do the installation.

CONCLUSIONS

We will see more and more complete software systems designed for specialized markets in the months to come. Both of the packages reviewed this month were designed by persons working in the fields for which they were created. The end user can be confident that the programmer at least understood what is needed for his business. Even most custom microcomputer packages are not designed by programmers familiar with the business in which the program is to be used.

Both the CAR/B system by A.S.I. and the GCP system by Microtex were very carefully thought out and perform well. The GCP package is designed to be easily installed by most dealers or experienced micro users and should find a very large market. The CAR/B system still requires a great deal of work during installation by a dealer, and is only available through authorized dealers. Both packages are extensively field tested.

A.S.I. is located at 1642 S. Parker Road, Suite 300, Denver, CO 80231, and they may be reached at the number above. Microtex is located at 5925 Sovereign Drive, Suite 101, Houston, TX 77036, and they may be reached at (713) 780-7477.

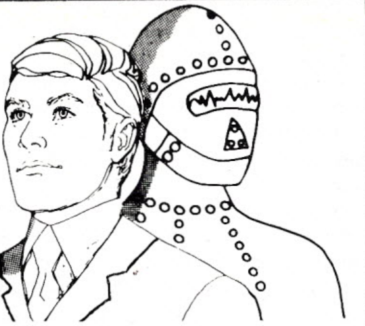
THE RATING CHART

This is a rating chart devised for use in this column. It is to serve as a rough guideline on several key points of interest to the business user. For an explanation of the categories, see the April issue of INTERFACE AGE.

	CAR/B	GCP
1. System Interchangeability	8	9.5
2. Program Interfacing	7	8
3. Maintainability	8	8
4. Documentation	4	9
5. Ease of Installation	2	8.5
6. User Lock-In	9	8
7. User Interaction	4	10
8. Input Error Checking	4	8
9. Error Recovery	9	7
10. General Design	7	10
Cumulative Total	**62**	**86**

All categories are on a scale of 1-10, 10 being the best.□

Carl Heintz and Bob Johnson will be alternating as authors of this column. Software vendors who are interested in having their product reviewed can contact Carl Heintz at 2540 Huntington Dr., San Marino, CA 91108. Bob Johnson can be contacted at 7228 W. Reno St., Rt. 5, Oklahoma City, OK 73108.



THE MIND REVOLUTION

By Merl Miller

Last month I proposed some basic ideas about an "attitudinal revolution," showing how man and machine have taken separate but parallel paths. This month I would like to take the discussion of these systems a step further and look at some specific examples.

The United States can be used as an example of how things change in a social structure. During the past few years there has been a variety of inputs into the political system that have caused enormous changes. The inputs are, of course, the various movements: civil rights, peace, women's rights, Proposition 13, etc. These movements are really the interplays of opposing forces: the government policies vs. some ideal, the status quo forces vs. the impetus for reform, caution or reform vs. change, etc. They all involve *robotic opposing forces in action* as their mode or principle of action. From a systems viewpoint and a social viewpoint some of these movements have been successful far beyond what normally could be expected.

It has already begun to show that majority interest, or the will of the people, can change government.

Think about the peace movement during the Viet Nam war. It was really impossible to imagine the people (rather than the leaders) stopping a major war and changing our national policy. It offers a beacon of light that major or global wars may be outlawed by ordinary people.

Next, consider the civil rights movement. It was as difficult to imagine that someday all people would be equal and, though it's not true yet, the possibility exists because the laws have been changed. There is a gleam of hope that all people will become socially equal with the same economic opportunity.

It has already begun to show that majority interest, or the will of the people, can change government. It has always been true on the local level and the impact is being felt on both state and national levels.

In recent years the United States has been criticized for lacking law and order, for being a weak leader and for not taking a hardline approach in diplomacy and military postures. These criticisms are really given without foundation. The United States is actually leading the world in a new revolution — the "attitudinal revolution." The government is no longer an open-loop machine that governs by brute force. Instead, it is becoming a closed-loop machine, or "intellectual robot," that *reflects* about its own performance. This revolution may make the U.S. a true democracy.

The current situation in Iran can serve as a model of what can happen when a feedback system overreacts. For years, Iran has been operated as an open-loop society with the populace having little or no input in government. As the people became better educated and more socially aware, the disenchantment with the government grew until the government was overthrown. Unfortunately, the change was too radical. The people may have exchanged one open-loop

CALL FOR ARTICLES

INTERFACE AGE is seeking articles on hardware and software, particularly concerning Apple, TRS-80, Atari and Heath computer systems. Other areas of interest include medical and educational applications, special functions of microprocessors, telecommunications and video graphics, both on screen and hardcopy.

The payment rate for articles ranges from $20 to $50 per published page. Pieces describing company projects or products will carry the company byline, but no payment is offered. Submittals should include an abstract, outline and stamped return envelope.

Manuscripts should be typed, double spaced with one-inch margins. Minimum length is four pages, unless programs are included. Photos should be numbered and have a brief description attached. Tables, listings, etc. should be on separate pages. Computer listings should be printed using a new ribbon to assure good reproduction. Authors are requested to submit a statement of their background and expertise.

The publisher assumes no responsibility for artwork, photos or manuscripts. No acknowledgement is made unless accompanied with a stamped return envelope.

For article submission or further information, contact Carl Warren, Editor-in-Chief, INTERFACE AGE Magazine, 16704 Marquardt Avenue, Cerritos, CA 90701. Please do not phone for information about submissions.

system for another. The essence of any closed-loop system is that it works with small but significant changes.

These are all principles that can be applied to any operating system. For instance, in Capede's classic *R.U.R.,* the robots overreact to the system and provide the means to their self-destruction.

But where does this leave us in our discussion of robots? What can we expect to develop and what kinds of safeguards should we take?

Here's an interesting concept — the broadminded robot. The robot is a relatively simple system model. Using this model, with the aid of simple algebra, we can mathematically prove that this great scientific creation has many human qualities, e.g., it can be broadminded, even tempered, agile, stable, visionary, resolute and reliable. There are a lot of other implications. For example, if a robot receives a "command," nature does not allow the effect or response to be identical to the command. Rather, the effect only approximates the command and it has both positive and negative overshoots. Doesn't this sound rather familiar in light of what we've discussed before now?

A robot is really the creation of numerous contributors. Partly because it is the cumulative effort of a large number of contributors and partly because the engineers and applied scientists are quite sloppy in their documentation, almost no engineering textbooks give references or credit to the creators of the robot.

People who design robots assume the design information is a common body of knowledge. Even though robots are to be used as the workhorse of industry, few people use the term. Instead, they use a variety of terms such as "feedback circuits," "feedback system," "automatic control system," and "servomechanism." They seldom think of robots in human terms, but of optimal operations. I believe we can assign some human qualities to a robot that can be mathematically derived from the robot's simple systems model.

How do you feel about this? Is the robot a cold, calculating machine similar to a typewriter or is it something like C3PO? Is it possible to have machine intelligence similar to biological intelligence or are we forever stuck with 11101 + 10111 = 110100? And, if we are, so what? We don't know how the human mind works, but who is to say it doesn't operate in a similar fashion?□

 CIRCLE INQUIRY NO. 54

R.H.D. real handy data

By Robert H. Distler

BUILDING ON THE SCHEMA

In the two previous issues, general schema to be used in the construction of the database were developed. This month, the subschema or how the application will interface with the general schema will be discussed, in relationship to the format of the data within the fields.

The point is determining how detailed an examination you will be making of your data, and the smallest detail you will be looking for.

I'll start with the two easiest fields to work with of the eight described last month. These are the five digit zip and the two-character state abbreviation.

The zip code field, Figure 1, contains in all cases five numeric characters. As a result it becomes unique to the record because it may be treated either as a numeric field or a string depending upon the desires of the application programmer. This field may also be considered a sub-key to the record since there is a high probability that you will want to sort the record based on the contents of the zip code field.

```
! 9 ! 0 ! 5 ! 0 ! 5 !
```

Figure 1. Example of zip code field with five sub-fields.

The input variable for this field may be defined, again depending upon the programmer's desire, either as Z for an all numeric field input or Z$ to indicate a string input. The BASIC program can also be constructed in such a manner as to check for a valid input based on the length of the field. It may be desirable to use the string input due to routing numbers used in other countries such as Canada that use both numerics and alpha characters in the number. However, for our purposes we will assume that all zip codes will be five numeric characters and will contain no alpha inputs.

The next field is the two character state field, Figure 2. This field will always contain alpha characters and be no longer than two characters. Even though the application may print the final result as CA., the period (.) is not entered at the time the record is built. This is done to cut down on the number of key strokes it takes to enter a record and to conserve memory, a period takes as much space as any other letter.

```
! C ! A !
```

Figure 2. An example of the two character state field.

This field may be defined in the BASIC program as S$. Again the program can check for length of the field and if desired check a state table for a valid state input.

The phone field is similar in nature to the zip code field in that it contains only numerics consisting of 10 digits. The first three sub-fields contain the area code, but for local areas may be filled either with zeros or the local area code by the program. For this situation, the program will look at the first three sub-fields, determine if they are blank and either fill them or leave them blank — it becomes dependent upon the programmer.

The actual phone number begins at sub-field 4, and contains no spaces, periods, dashes, or other punctuation. When the program prints the field, it will add the necessary enclosures and dashes, i.e. (213) 926-9544. The construction of the phone field is shown in Figure 3.

! 2 ! 1 ! 3 ! 9 ! 2 ! 6 ! 9 ! 5 ! 4 ! 4 !

Figure 3. Example of phone field with 10 sub-fields.

Like the other fields, this field may be defined as P in the BASIC program. It may be desirous to treat this field as unique and as a possible sub-key for sorts on phone fields.

Now let's look at the fifteen character city field, Figure 4. The first character of the city name will always be at position #1. Any remaining characters after the city name will be filled with spaces. The name will be allowed to contain spaces, i.e. San Diego. If a name requires abbreviations, periods will be allowed. The end of this field is padded with spaces, differing from the zeros used to pad the beginning of the phone field. This padding is to maintain our fixed length field and data within that field in known format.

The two name fields will be allowed to contain capital letters, lower case letters, periods and punctuation. However, last names like "MacDonald" will not be separated. Suffixes at the end of names, i.e. Jr., Sr., Co., will be followed at all times by a period. This will be used later to determine last names within a field. All unused character positions in these two fields will be filled — padded — with spaces.

! C ! E ! R ! R ! I ! T ! O ! S ! ! ! ! ! ! ! !

Figure 4. Fifteen character city field with example of padding.

The twenty character address field becomes more complex because it can contain both alpha and numeric data. However, by using predefined rules we can simplify the complexity. The street number will always start at character position #1, and end with a space. Immediately following this space an optional street direction letter may be inserted. This letter, if used, must be followed with a period. A space is optional between the period and the first character of the street name. The name may contain both numeric and alpha characters. For example, 10th would be entered "10th" without a space between the 0 and the t. The street name is ended with a space. If a suffix such as Ave. or St. is used, it must end with a period, so that later the street name can be found within this field. All unused character positions in this field will be filled with spaces.

The last field is most important of all. This field must be an absolute unique identification for each record. No two records can have this field being the same. For our purposes, we will limit this field to five digits and a check digit. Keeping this in mind we will be able to create the customer identification number. The five digits can be 0 through 65500. The check digit will be generated by the system at the time a new customer record is entered. From that time on the system will require all six digits to be entered when creating a customer number. The first five digits are used to test the check digit, thus reducing errors in the 'id' number at entry time.

THE WHY OF THE CONSTRUCT

The zip and city fields will primarily be used for mailings to take advantage of discounts for bulk mailing. These fields can also be used as sort keys to define a sales area or to monitor sales performance for a sales report — but more on this as our schema and sub-schema definitions grow. More importantly the sorted data can be used for selective advertising mailing.

The reason for all the rules in the address field is to allow for the location of an address by: street name, and type. Plus it gives the capability of resolving the lists to specific street numbers if desired. Definitive resolution, such as this, would find application in a real estate office to announce an open house within a given area, or to set up a sales campaign within a specific area.

The remaining fields provide the ability to sort data into more useful formats for specific day to day needs. Even though the customer name and company name fields are the same with regards to the length and rules applied to the data, they are unique in themselves. This uniqueness gives the ability to establish ideal keys to sort the data into two distinct lists.

Unfortunately, all the fields mentioned so far do not give the absolute database resolution that is necessary. Resolution of a database can be compared to a landscape picture. How much detail can you see within the picture: Can you see a tree? Can you see the leaves on the trees? And so on. The point is determining how detailed an examination you will be making of your data, and what is the smallest detail in them that you will be looking for.

These points of resolution of the data are satisfied by our customer number. The customer number is unique to each record within the data file. This number allows us to directly locate a single record. This means, of course, that the application can locate a given record check for specific detail and manipulate it. Also the resolution provided by the customer number or, if you will, record locator, gives the cross reference needed to link other databases or schemas.

Next month, I will discuss the input format of this database so that you can begin building it up on the media.☐

Do you have a specific question regarding data handling or data base management? Then write to R.H.D. care of Robert H. Distler, P.O. Box 6376, Oxnard, CA 93031, or call (805) 487-7422.

Ask Eliot Janeway

Interview by Terry Costlow, Assistant Editor

The price of energy is affecting every industry. Do you think Carter's oil plans will change anything?

JANEWAY: He has no oil plans. He's just acquiescing. I'm glad you asked the question, because it's the basic question. Oil is the lowest common denominator of all activity. When it goes up, everything goes up.

In the past few months, you have been a supporter of the uranium-bearing South African gold stocks. Do you think the nuclear problem in Pennsylvania will affect these stocks?

JANEWAY: I think it's a phony.

You think the problem is being blown out of proportion?

JANEWAY: Oh, sure. The Nuclear Regulatory Commission has a vested interest in proving that nuclear is unsafe.

Why would they want to say it was unsafe? They'd be putting themselves out of work.

JANEWAY: I don't know that it would put them out of work. It would put them in the driver's seat. They're part of the environmental lobby. They need to justify their arbitrary action of having shut down five nuclear plants in March, just when the oil ripoff was starting. So the Pennsylvania accident, the Middletown accident, was providential from their viewpoint.

Do you think it will slow down the growth of the nuclear industry?

JANEWAY: There's no growth in the United States. The only growth has been overseas. The rest of the world is going nuclear by leaps and bounds and we're stranded in a dead stall, doing nothing about it.

You have a real test now of whether nuclear is safe or not. Countries with technology that can't match America's — from Korea to Finland — are going forward, led by Japan. If any country could be expected to be gun-shy about nuclear, it's Japan.

But they're going forward. Libya's going nuclear; Russia's doing it. So my point to you is that if the rest of the world makes it work and we don't, then clearly **we** have a problem.

One of the concerns people there seem to have is that the people who run the nuclear plants are not being completely honest with them. . .

JANEWAY: Do you want a scoop of great importance to the computer industry? There's been a very serious lapse on the part of the nuclear industry. The lapse is that there has been no standardization. So if you start locating this plant and then ask what went bad, you have to go through 16 bluebooks to identify the part that went wrong. If something goes bad on a 727 airliner, or a 747, you know immediately what to do about it. It's all standardized. But nuclear plants aren't standardized.

Now maybe the manufacturers will complain that it's the fault of the anti-trust laws. I don't know what they'll say. But there should have been an alert requiring standardization. And by God, that's what you've got these regulators for! The regulators have their noses in everybody's business; they're telling everybody what to do and what not to do; but the most rudimentary standard of regulatory precaution has not been met.

Could you tie that into your comment about the computer industry?

JANEWAY: I tie that into the computer industry because obviously it's a challenge and a market for the computer industry to standardize all these plants.

They've got enough problems of their own trying to standardize the computer industry.

JANEWAY: Well, that's right. Tell them, let them read about it.

I think it's bullish for the nuclear industry to have a confrontation with the regulators.

Who do you think will win in this battle?

JANEWAY: The industry.

You don't see any effect on the uranium-bearing gold, because of this?

JANEWAY: No. From an investment standpoint, what's bad for uranium is bad for the dollar. What's bad for nuclear power is bad for the dollar. What's bad for the dollar is good for gold. If there's going to be a retardation of nuclear expansion, and if that is bad for uranium, mines producing both commodities will still benefit because gold will benefit. Gold has no value. Gold swings with the market.

Today the stock market is down $9, I guess on the evidence that there's no government. So that's bad for the dollar, which is good for gold.

Switching to politics, do you think that Jimmy Carter has much of a future?

JANEWAY: He's gone. He's a loser. He's the first lame duck in history to spend two years in office quacking.

Who do you think will be the Democratic candidate in 1980?

JANEWAY: Kennedy will run. He can have it if he runs. And my impression is that he will want it.

And on the Republican side, do you think Ford will try again?

JANEWAY: No. He'll never be nominated. I think Conally's running away with the Republican nomination.

Do you think his ties with the Nixon administration will be much of a drawback?

JANEWAY: It's a liability. But on the other hand, you can't have it both ways against him. If you say: "You worked for Nixon," which I consider a liability — I'm a Democrat — you can't then say: "You're not a Republican." The answer is: "If I worked for Nixon, I've sure paid my dues as a Republican. That was some sacrifice to pay for showing that I'm a Republican."☐

Questions for this column should be sent to Ask Eliot Janeway, P.O. Box 1234, Cerritos, CA 90701. One question per letter, please. Personal replies are not possible.

The comments expressed in this column do not necessarily reflect the views of this magazine or the microcomputer industry.

—The editor

THE INVENTOR'S SKETCHPAD

By Roger Garrett
Northeastern Regional Editor

A PROPELLED DROPLET PRINTER

Ink-jet printing, as seen in IBM's TV commercials, is a relatively new form of printing in which tiny droplets of ink are 'fired' at the paper to form dots which, when fired at the appropriate positions, form the desired characters. One of the many advantages of this form of printing is that with simple changes in the character-generating ROM any form or styles of characters may be produced at camera-ready quality. This means the quality of type that you normally see in magazines.

I do see a problem with the current state of ink-jet printing, however. Of the two currently available forms, that of 'ink-on-demand' and what can best be described as 'steady stream,' both require that the printing element be moved back and forth across the page, just like conventional dot matrix printers. The 'ink-on-demand' form fires the droplets only when needed to produce a dot. The 'steady stream' approach fires droplets continuously but deflects into an ink reservoir those droplets which are not needed.

Wouldn't it be nice if the printing element could be held stationary so that the only part that moved would be the ink droplet itself. Such a device would print faster and require less maintenance. . .

This latter form, which was the first to be developed, has the added problem that the unused ink must be cleaned by a special filter and recycled by a special pump back to the ink-jet stream. The major problem, however, with each of these methods is that the printing element, the device that actually shoots the ink at the paper, must be moved back and forth across the paper.

Wouldn't it be nice if the printing element could be held stationary so that the only part that moved (other than the normal movement of the paper) would be the ink droplet itself. Such a device would print faster and require less maintenance due to fewer moving parts. And here is how I believe it could be done.

FIGURE ONE

We take a piece of tubing cut to the width of a piece of paper. A slot is cut the entire length of the tube. Inside the tube we position a metal plate opposite the slot. This plate we will call the *electrostatic*

deflection plate. Around the outside of the tube we position individually addressable electro-magnetic coils, each of which has a slot cut into it to correspond to the slot in the tube. We will call these *propulsion coils.*

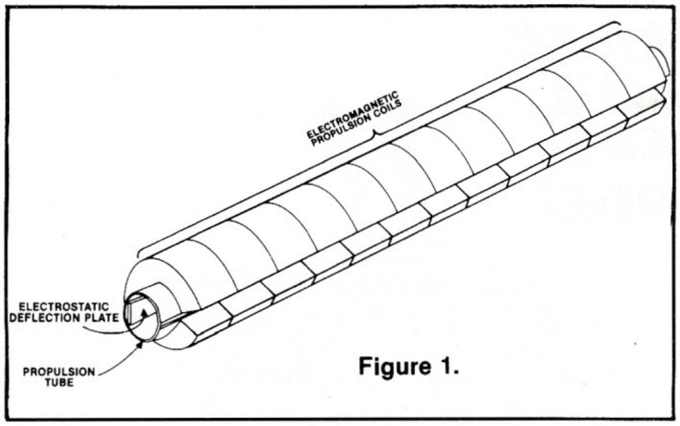


Figure 1.

FIGURE TWO

At the open end of the slot we mount a platen which moves the paper past the slot. Also shown in this cutaway end view is a single ink droplet in the center of the propulsion tube.

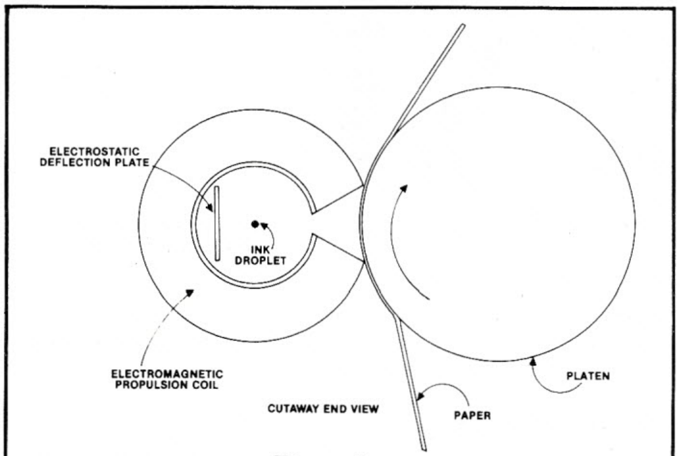


Figure 2.

FIGURE THREE

Now we are looking at an overhead cutaway view with the platen removed. The cutaway portion of the propulsion coils are shown cross-hatched. The open slot in the propulsion tube and coils faces downward.

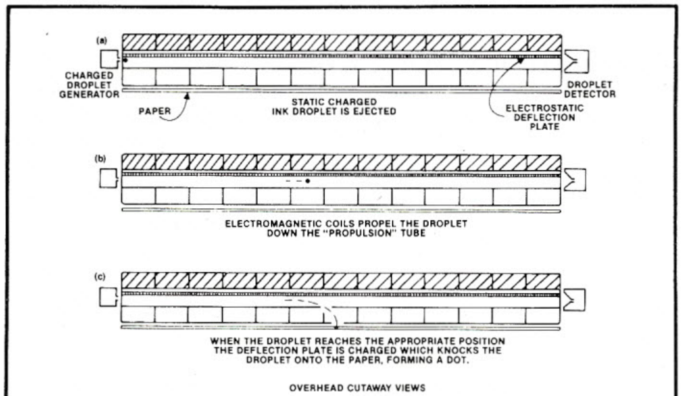


Figure 3.

In Figure 3a we see that a *charged droplet generator* has been added at the left side of the propulsion tube. This device is very much like the steady stream ink-jet printer in that it 'fires out' electrostatically charged ink droplets, yet it is also like the ink-on-demand

the marriage of the electric pencil™ to TRS-80* is still going strong!

TRS-80 is a product of Radio Shack, Div. of Tandy Corporation

NOW! The Word Processor you've asked for!

The well known Electric Pencil, which has been in use for over two years with most popular microcomputers, has also been available for use on the TRS-80. We are now proud to announce immediate availability of The Electric Pencil Disk Version for the TRS-80, featuring the same fine capabilities that have made the Electric Pencil so popular. Full price for the TRS-80 version is only $100, and for the TRS-80 disk version, only $150.

the electric pencil

—a proven word processing system

Extremely easy to use, The Electric Pencil lets you produce mailing lists, business forms, large numbers of original correspondence, camera-ready copy for printing . . . all on your TRS-80.

The Electric Pencil is a character oriented word processing system, which means you have maximum freedom and simplicity in the handling of text. No carriage returns or word hyphenations are necessary. Line formatting is done automatically by The Electric Pencil.

The Electric Pencil combines features found in more expensive systems with many capabilities you won't find anywhere. Right margin justification, page numbering and titling, and many combinations of line length, page length, and line and page spacing are always under your control. Text editing capabilities eliminate guesswork when inputting. The text appears on the video monitor as you insert it, and you can view text whenever you wish using the bidirectional variable speed scrolling feature. You can insert, delete, or relocate any text using simple keyboard commands.

The Electric Pencil has been designed to work with both Level I (16K system) and Level II models of the TRS-80, and with virtually any printer you choose. Level II owners will want the TRS-80 disk version, which is supplied on cassette. Transfer of the Electric Pencil to your own disk is as simple as entering a command.

Upgraded Versions

The Electric Pencil TRS-80 Disk Version is fully interactive with the DISK READ, WRITE, DIR, and KILL routines of TRSDOS 2.1. Any version of The Electric Pencil may be upgraded to accomodate new equipment. For instance, if you are already using The Electric Pencil and have added a floppy disk to your TRS-80 system, simply return the original cassette to Michael Shrayer Software, along with the price difference between your present Electric Pencil and the upgraded version, plus $15 for handling.

Demand a Demo from your Dealer!

Your dealer will have a manual and descriptive literature for you to see, and can demonstrate The Electric Pencil to you, on one or more of his demonstration microcomputers. Look the manual over carefully, and note the explicit instructions which lead you easily through The Electric Pencil operation. The manual was produced using The Electric Pencil which will enable you to see its many capabilities for yourself. Then try it out on your dealer's demo unit . . . most any microcomputer will do. If, for some incredible reason, he's unprepared, demand it! or write:

mss

MICHAEL SHRAYER SOFTWARE, INC.
1253 Vista Superba Drive
Glendale, CA 91205

printer in that it fires droplets only when instructed to by the controlling hardware (or software).

At the right-hand side of the propulsion tube there is a *droplet detector.* This is *not* a droplet recycler. When an ink droplet reaches the detector its presence may be detected (probably by a simple LED/light sensor arrangement), but it is not recycled. In its simplest form it would be a sponge which would be discarded when full. But that is quite all right because very few droplets ever reach the detector.

Let's start at the top with these drawings (Figure 3a). At this point an electrostatically charged ink droplet has been 'fired' out of the droplet generator.

The controlling software now takes over (3b) by sequentially energizing and de-energizing the individually addressable propulsion coils. It is similar to a linear accelerator which is used in atomic research to move electrons and other charged particles. The ink droplet moves down the propulsion tube through the properly controlled magnetic fields of the propulsion coils.

In Figure 3c the droplet has reached the position where we want a dot placed on the paper. The propulsion coils are turned off and the deflection plate is immediately charged with a strong electrostatic charge of the same type as the droplet. This deflects the ink droplet towards the paper and a dot is accurately positioned on the paper.

FIGURE FOUR

Here is a perspective view of a typical setup for this system. Notice that the paper is printed one line at a time. This is similar to the way in which a CRT monitor works. Assuming that each character to be printed is actually made up of a series of dots, the top row for all characters on a given line is printed first. The paper is then rolled up one dot's height and the second row of dots for all characters on the line are printed.

It might even be possible to have more than one ink droplet traveling down the propulsion tube at the same time simply by dividing the deflection plate into several individually addressable sections.

Continued below

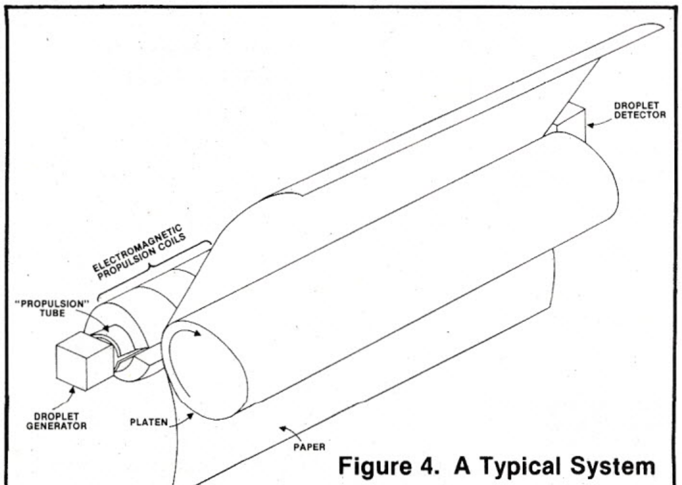


Figure 4. A Typical System

The function of the droplet detector is to periodically check to ensure that droplets are properly traveling down the propulsion tube and at the correct velocity. Every once in a while the controlling software will fire a droplet down the entire length of the tube and test to make sure that it gets there in the expected amount of time. If it doesn't get there, then it knows that it is out of ink or there is a fault with the droplet generator. If the drop is too slow or too fast, it will know to compensate by charging the deflection plate appropriately.

The final result, then, is a printer with only two moving parts: the platen and the ink droplets. There may, of course, be problems with propelling the droplets for any great distance. The width of normal computer print-out paper (14") might be too much. A possible solution might be to have a propelled droplet printer for each half of the paper width which, of course, would double the printing speed.

Such a device might also be utilized on hand-held calculators, providing them with simple printing capabilities and replacing the thermal printers which require special paper.☐

The author can be contacted at: Inventor's Sketchpad, 16 Grinnell Street, Jamestown, RI 02835.

Now PASCAL
For Greater Power Potential With Your
TRS-80®

PASCAL, the powerful, general purpose language developed for large and complex programs is now available for your TRS-80. This new, Industry standard language avoids many errors normally encountered when using older languages.

Although it is relatively easy for beginners to learn and use, it provides added potential to cover most problems of practical programming. The UCSD PASCAL Compiler allows long variable names, and block structured control statements.

The FMG/UCSD PASCAL System opens a new generation of value for your TRS-80.

Package Includes
- Operating System
- Screen Editor
- Z80 Macro Assembler
- Debugger
- Pascal Compiler
- Utilities and System Reference Book

$150.00

Requires 48K System with 2 Drives

Available without Macro Assembler Linker and Debugger $100.00

Available PASCAL Books		
"Microcomputer Problem Solving Using PASCAL"	*By Dr. K.L. Bowles*	$ 9.95
UCSD Pascal Reference Manual	*Version 2.0*	$19.95
Pascal User's Manual & Report	*By I.C. Jensen & N. Wirth*	$ 9.95

CBASIC
A commercially oriented BASIC compiler-interperter designed for use with the CP/M disk operating system. The compiler features
- ☐ Variable Length Files
- ☐ Double Precision Numbers
- ☐ Optional Line Numbers "While . . . End"
- ☐ Structural Group
- ☐ Long Variable Names (up to 31 characters)
- ☐ Multiline User Defined Functions
- ☐ 98 Page Manual

$99.00
(Manual only *$25.00*)

TEXTWRITER II
Textwriter II is a text formatting program that prints files created by an editing program. Form letters can be printed from a mailing list file and personalized with automatic name and address insertion anywhere in the body of the text.

$130.00
Manual *$25.00*

FORTRAN
Comparable to compilers on large mainframes and minicomputers. All of ANSI Standard FORTRAN X3.9-1966 is included except COMPLEX data type. Therefore, users may take advantage of the many applications programs already written in FORTRAN. Package includes:
- ☐ FORTRAN Compiler
- ☐ Macro Assembler (Z80)
- ☐ Linker
- ☐ Library
- ☐ Lib Manager (Not in TRS-DOS version)

Price *$350.00*
Manual *$25.00*
(Specify TRS-DOS or CP/M versions)

MAC MACRO ASSEMBLER
Compatible with new Intel Macro standard. Package includes complete guide to Macro Applications; MAC Diskette and Manual

$150.00

MACRO ASSEMBLER
A relocatable Macro Assembler for both 8080 and Z80 opcodes. Incorporates most big computer assembler features without sacrificing speed or memory size. Package includes:
- ☐ Comment Blocks
- ☐ Full Intel Standard Macros
- ☐ Lib Manager (not included in TRS-DOS version)
- ☐ Linker
- ☐ Library

$219.00 (Included in FORTRAN)
Manual *$25.00*
(Specify TRS-DOS or CP/M versions)

TEX™ TEXT FORMATTER
Powerful text formatting capability using CP/M Editor. TEX Diskette and Manual *$125.00*
Manual *$25.00*

CP/M OPERATING SYSTEM
Editor, Assembler, Debugger and Utilities for 8080 and Z80 Systems. Up to four floppy disks. Package includes:
- ☐ CP/M System Diskette 5¼"
- ☐ CP/M Features and Facilities Manual
- ☐ CP/M Editor's Manual
- ☐ CP/M Assembler Manual
- ☐ CP/M Debugger Manual
- ☐ CP/M Interface Guide

$150.00
(Set of 6 manuals only . . . . *$25.00*)

SID™ SYMBOLIC INSTRUCTION DEBUGGER
- ☐ Symbolic Memory Reference
- ☐ Built-in Assembler/Disassembler
- ☐ SID Diskette and Manual

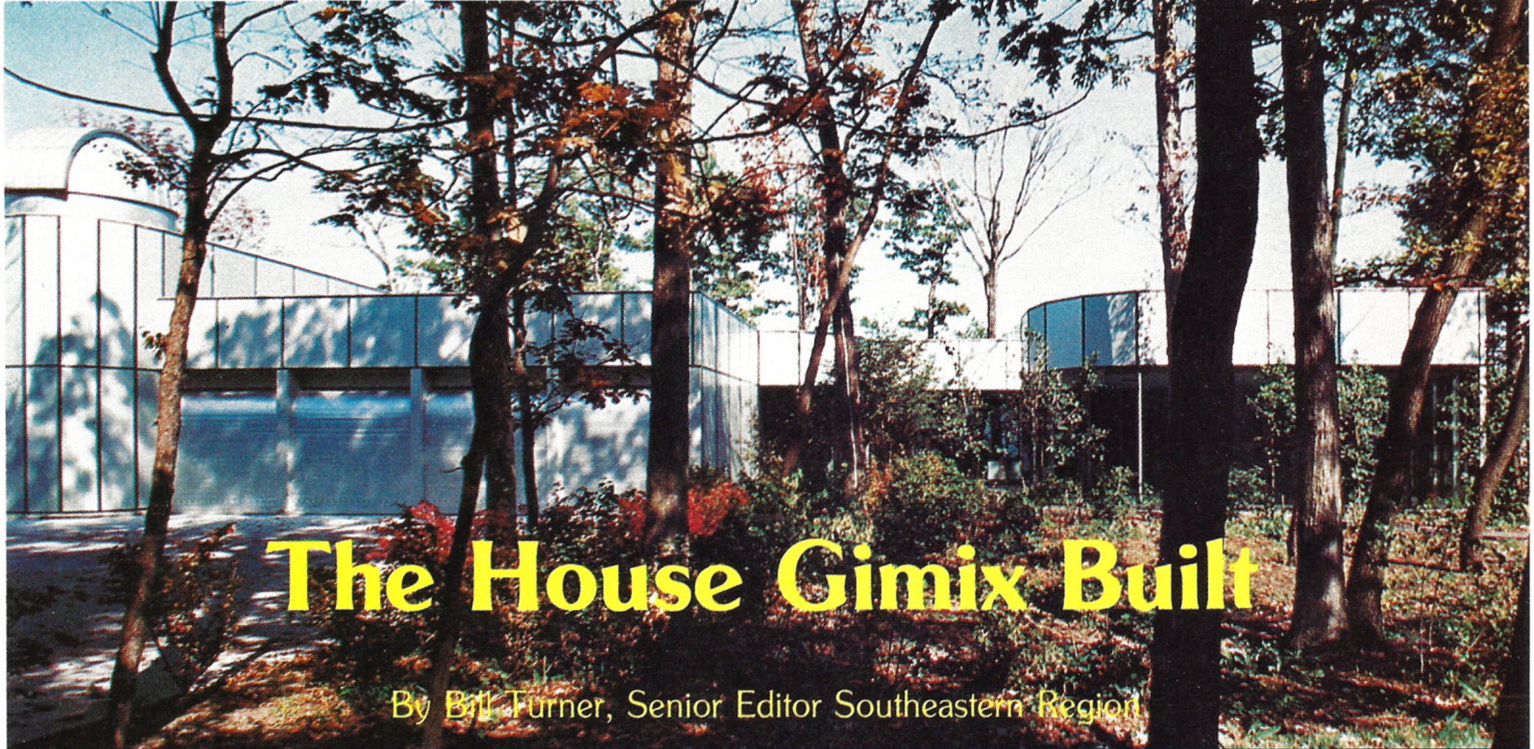
$125.00
Manual *$25.00*

Call or Write for Complete Information

FMG for HIGH LEVEL LANGUAGES
BASIC FORTRAN COBOL and now PASCAL

CP/M is a registered trademark of Digital Research Corp.
TRS-80 is a registered trademark of Radio Shack.

FMG CORPORATION
P.O. Box 16020 • Fort Worth, TX 76133 • (817) 294-2510



The House Gimix Built

By Bill Turner, Senior Editor Southeastern Region

From what was intended to be an elaborate garage door opening device, a Chicago resident has ended up with a house that can be run almost entirely by computers from anywhere in the world.

This surprising combination of the garage door opener, a computer and a telephone paging system came about in part through the work of employees of Gimix, Inc., a company that specializes in manufacturing telephone devices and microprocessing equipment that is based on the Motorola 6800 and the SS-50 bus. The product is called the "house control system."

The house control system is actually the commercial adaptation of a computerized control system that has been controlling several Chicago area homes for several years.

Architect Stanley Tigerman was selected to design the house, which was to be "different." Just how different was a source of concern for the owner's wife. He wanted a futuristic MACHINE IN THE GARDEN.

She was more interested in a "white brick country villa," and was less than enthralled with the glass and stainless steel cladding that was proposed.

Eventually, though, everybody agreed to the duller pewter-like finish of aluminum in place of stainless steel. The glass and aluminum structure nestled beneath the snow covered trees at the end of a long curved driveway is truly impressive. The only regret of the owners is that perhaps less glass and slightly lower ceilings might have been better.

The house sits atop a 1.5 acre cliff overlooking Lake Michigan. The floor plan derives its unusual configuration from the decision to have rooms overlooking the lake, and the desire to provide space and privacy for each of the five family members. There are several unique features, such as the two walkways that run the full length of the house along the windows on the east and west walls. This eliminates the necessity for other house members to interrupt guests while "passing through." The house also has a radiant-heated indoor pool and an observatory where the owner can enjoy one of his many hobbies — star gazing.

The owner, who prefers anonymity, took a much larger interest in the planning and construction aspects than normally occurs. In fact, he acted as the general contractor for the prefab custom house and included extra features such as installing guidance cables in the floor for a computerized serving cart. Much to my disappointment, and also to several recent visitors from Japan, the cart has not yet been assembled.

The house was built using techniques very similar to those used in constructing a warehouse. The roof is supported entirely by joists that are themselves supported by columns built into the outside walls. To use the more technically correct architectural terms, the house is described as ". . .being designed using a structural mullion system of lally columns on 5 foot centers, which carry 35 foot long open web steel joists. The interior space is a structurally uninterrupted space approximately 35 feet wide and 100 feet long. . .".

This, for "non-architects," simply means that none of the interior walls hold up the roof, and the 35 by 100 foot interior space can be redesigned without tearing down the outer walls or the roof.

One of the many features of the house is its four-car garage with separate roll-up garage doors. And for a house overlooking Lake Michigan, that can be handy in mid-January. Apparently, the owner seems to like warmth, so he tried to order a radio controlled garage door opener.

No problem, as long as you don't want a radio controlled garage door opener that will control each door separately and independently from the others.

One suggestion was to use four separate systems. The "engineer" who suggested this overlooked the problem of having the correct transmitter in the right car. When asked how to solve this problem, the "instant" solution was to buy a separate transmitter for each car.

But with four cars and four different systems. . .that makes 16 transmitters. Besides requiring a very large glove compartment, eventually someone would end up with two or more transmitters for the same door. Where would the correct one be? Clearly, this was not a very acceptable solution.

The pertinacious owner was not about to give in. Armed with the knowledge that model airplanes could be controlled by a four-channel radio control system, he went looking for someone who could build a garage door opener. Eventually he was introduced to a young electronic wizard, Robert Phillips. Bobby was subsequently hired to design, build, and install a four-channel radio controlled garage door opener for the house.

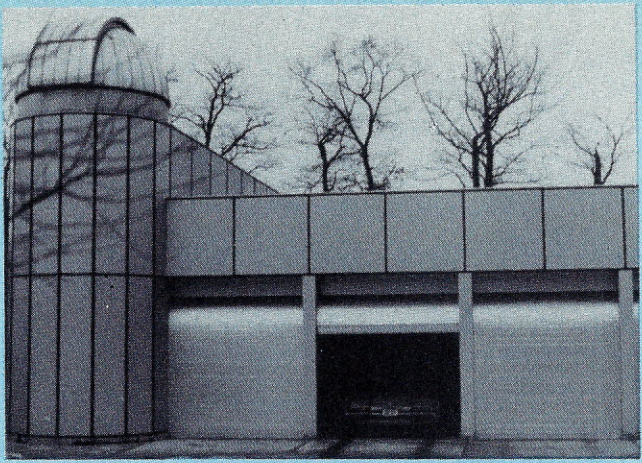
The owner may have ordered a garage door opener, but what he received was a system capable (when expanded) of controlling the entire house.

As a result, the small hand-held radio transmitters with their telephone-styled pushbuttons can actually be used to control anything that is connected to the house control sys-

tem. All of the house's normal "environmental control" systems are connected to the computer system that Bobby designed and installed. These functions include the house heating systems, the air conditioning system, and lighting systems. In addition, the stereo, TV, some kitchen equipment, the security systems, etc. are also connected to the house control system.

According to Bobby, most control systems place very expensive receivers inside the equipment to be controlled, and make the transmitters extremely simple. Bobby feels that the transmitters should contain more logic, so that the receiver in each device can be the simpler (and less costly) device. This would allow a single transmitter to control many different devices, rather than having to have a special transmitter and receiver pair for each device.

The 21 telephones throughout the house are also multi-function devices. Each has a four-digit readout device that



Micros Join in the Conservation Battle

By Terry Costlow, Assistant Editor

While energy prices and technological advances race each other into the 1980s, a Southern California businessman is using one to defeat the other. Bill Mandl, president of Hometech Computers, has created an interactive computerized control unit to keep the cost of powering the home to a minimum.

In addition to managing the power consumption, the prototype home in the hills near Los Angeles features a security system, timing controls for a variety of equipment and priority scheduling of some other appliances. The system is designed for installation in any home costing $100,000 or more.

Mandl put his $7,000 computer system into the $350,000 three-bedroom home last December as construction was finishing and has since been testing the operation of his Intel 8085-based machine. By publication time, it is hoped that the test system will be removed and the company's new production-line model will be installed. That vital step comes when the house is sold.

An important detail for the prospective homeowner is the energy savings of up to 50% which the computerized system offers. The timing control unit manages a sunshade closing system by sensing the amount of light coming through the many picture windows. The heavy shades will be opened or closed, depending on whether the heating or cooling mode is operational.

"With the sunshade controller, we meet the standards for the state energy conservation tax rebate. That allows the buyer to get back up to $3,000 against the price of our unit. That's nearly half the cost of the computer. And with the energy savings on top of that, it's saving you money from the beginning," the electrical engineer says.

Water is conserved via sensors in the lawn which tell the computers to turn on the sprinklers only when necessary. A zone temperature system keeps heating and cooling costs at a minimum by circulating air when possible.

The computer also has a security system that turns on the lights, sounds a horn and, with an optional dialer, calls the police or a neighbor. When someone attempts to break in and all those systems go into action, the description of the area where the door or window is being opened is displayed on the CRT screen, allowing anyone at home to avoid the thief or go to the scene of the problem.

The security also hooks into smoke detectors for fire protection. As with the alarm system, it will tell the user the area of the alert. In addition, the zone temperature control is commanded to turn off all the fans and close the vents to slow the circulation of smoke throughout the house.

As with most power control systems, there is a priority shutdown for all appliances when the emergency reaches a hazardous point.

To facilitate operations throughout the house, there are RS232 ports in most rooms. The owner can plug in a terminal and monitor any room, change any of the timers or reprogram parts of the control package.

But to make the system easy to run, Mandl has made most of the controls self-supporting. The home can operate without manual adjustments most of the time. And when changes are desired, menus and complete programming instructions are printed on the terminal screen.

Because there is little need for reprogramming, there is only 1K of RAM or memory. The system uses 8K of ROM now, but there is room for up to 15K. The additional ROM will probably be used when food processing is expanded from timed coffee making and other simple chores now being done in the kitchen.

Hardware alterations and software filtering allow the computer to distinguish quick noises and power surges such as those that occur when a doorbell rings from legitimate commands. Each section of the control unit has specific safeguards so it is not affected by extraneous information. To keep on track, the system also goes through a self-recovering recycle every 24 hours. To avoid potential tampering, the computer lines are run independently from the electric lines.

While controlling the temperature, turning on the lights when the home is being burglarized and starting a cup of coffee when the alarm clock goes off aren't dramatic innovations by themselves, Mandl feels his computerized system will soon become standard in new houses because the functions work together.

"If you have other things like timers running different things, they still won't be interactive like this computer. By the time you get all the separate controllers and the priority shutoff and the phone dialer and the closing air vents and the rest, if you pile all that stuff on your house it would cost you more than our interactive unit," Mandl explains.□

continuously displays the time. Since all phones in the house are also used as computer input devices, the entire house can be controlled from any pushbutton phone, either in the house, or for that matter, in the world. (To control the house from outside the house, two things must be known — the telephone number and the security number used to access the computer.) The inside telephones are also part of a phone-to-phone intercom or a housewide paging system.

Even the radiant heating system in the indoor pool area can be remotely controlled. The owner emphasized very strongly, however, that no machinery that could be considered dangerous can be turned on from anywhere other than the immediate area of that device.

Any of the 200 ceiling lights (there are no floor or table lamps) can be turned on or off, in any combination and with a choice of brightness levels.

This allows a house occupant to "pre-program" any mood desired, and to cause the room to assume that "mood" with a single command. Any number of "moods" can be stored in the system and retrieved on command. Even after a "preset" mood has been commanded, individual lamps may be independently controlled to further adjust the mood.

A snow-melting system installed under the driveway is also controlled by the computer. The outdoor sprinkler solenoid control valve is also connected to the computer. This allows the owner to turn on the sprinkler system from inside the car as he leaves for work each morning. Any of a number of devices could be used to turn off the water, including the simple method of noting what time the water was turned on, and turning it off a half-hour later.

The house control system also makes it easier to conserve energy and yet not hamper or restrict the homeowner's lifestyle. The GIMIX house has two completely separate heating systems — a radiant heat system in the floor and a forced air heating system that blows warm air in from ceiling diffusers.

The radiant heat system has enough capacity to raise the house temperature 50-60 degrees above the outside temperature. In the fall and spring the boiler water temperature is set to 120°. During the winter the water temperature is set to 140° to provide extra system capacity. The burner is a modulating gas valve, which provides only enough heat to maintain the water in the boiler at a constant temperature. The garage and utility area are on a separate radiant heat system, which is currently set to maintain the normal temperature in those areas at 40 to 50 degrees.

The forced air heating system is currently broken up into five separate zones, each controlled by its own thermostat. The thermostats are connected as part of the computer OUTPUT devices.

The owner emphasizes safety, and he feels that to tamper with any part of the furnace control system is courting disaster. As a result, the computer merely controls the power to the thermostat, never the output. This way if the computer system should fail, the worst thing that could happen is either the thermostat controls the heat in the normal way or that the heating system is totally shut down.

The safest method for temperature control is also the most economical. To have several preset temperatures, the owner selects one or several thermostats, Again, the "fail safe" feature is that should several be selected at the same time, the room temperature would not rise above the setting of the highest thermostat.

When the house is unoccupied, the forced air system is totally shut down, leaving only the radiant heat system active. This allows the house temperature to fall to approximately 60°, conserving heating costs. The computer can be commanded remotely through the use of a standard pushbutton phone, making it possible to switch the forced air system on just before leaving work. This provides the best of two worlds, energy conservation and a nice, warm house to come home to.

Several security devices are also implemented; fire and burglar alarms, electric eyes in the driveway and entrance walkway. The house also has a closed circuit TV that shows approaching visitors. As the power to the closed circuit system is also controlled by the computer, it is automatically turned on whenever someone enters the premises. All of the security systems devices are direction sensitive — they are triggered by approaching vehicles or people, but not by anyone leaving the premises.

In the GIMIX house, a car coming up the driveway will signal the house, and if it is dark out, the driveway lights will go on automatically. This is also true of the garage lights — at night they are turned on when the garage door is opened.

Inside, lights and other equipment can also be controlled by the computer. The house really controls the wall outlets that various equipment is plugged into, rather than to modify the equipment for remote control operation. This proved to be the undoing of a very clever idea for controlling the owner's sons' stereo equipment.

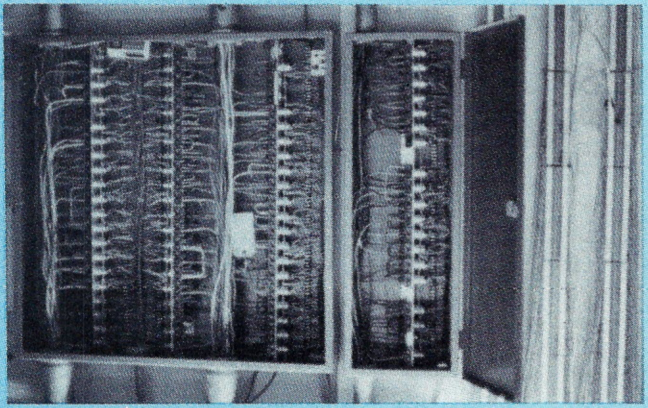
The three teenage boys each had their own stereo. Old "dad" figured that when the sound got too loud, he could key in the "off" command for the wall outlet that the offending unit was plugged into. This worked for about a week. One son started playing the stereo loud, then Number 2 joined in, and finally all three boys were playing the stereos full bore. Dad punched the off button for the wall outlet — nothing happened. He turned off another wall outlet — still nothing. He commanded the computer to shut off the power to the room — still nothing happened.

He figured that he had a computer failure, and started for the boys' rooms. As he approached, he noticed a long extension cord coming. . . They knew that he would try to shut the power off. So they got power from the kitchen via a long extension cord.

CONTROLLING THE UNIT

The computer receives its commands in a number of ways: by pressing small switches throughout the house or by dialing a code using one of the telephones, or even by using one of the garage door remote control transmitters.

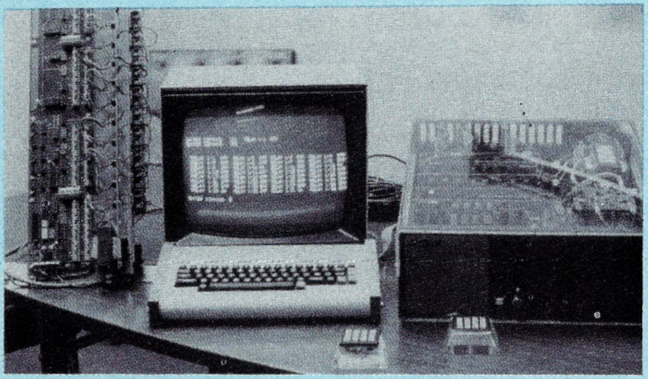
Each room can also be controlled manually through the use of small pushbuttons located on the doorjam of each room. These buttons control the remote control relays directly, and do not require computer assistance. This is done for safety, in case the computer should ever fail.



Pushbuttons on any of the house's 21 "control panels" or on the portable transmitters direct the computer by means of a three-digit code. As currently programmed in the prototype system, the computer can understand up to 999 different commands. The system is set up so that the first button in each series of three specifies a different section of the house. The second digit asks for a general function, such as turning on all the lights. The third has a more specific function. For instance "123" could mean turn on the third ("3") light ("2") from the window in the living room ("1"). "423"

could mean the same thing, but in the family room ("4").

If a command starts with a star ("*"), it identifies not a command for a specific device, but instead it identifies a string of commands for the computer to follow. "*10" could mean turn on all the living room lights according to the preprogrammed sequence "121,122,123".



The owner of the house said there were some unexpected savings that he attributed to the computer. Because room light controls are readily at hand for any room, the lights are seldom left burning when not in use. Lighting levels themselves are no higher than needed. When the computer controlled lighting is first switched on, it is turned on at a low-brightness level; for more light you must press the button again. The end result is that bulbs last longer, and less power is consumed.

Several other techniques were used to conserve power. In the laundry room, for example, the wall outlet for the iron is on the same circuit as the ceiling lamp — no light, no iron. Apparently his wife seldom leaves the room lights on, but will forget and leave the iron on.

The dressing area in the master bedroom is lit by fluorescent lights. The owner had four tube strips installed, but had two tubes in each fixture wired to one circuit, and the remaining two wired to a separate circuit. This way when a tube burns out, all he has to do is switch the lighting control to the other circuit. Now he can change the burned out tube when he has time.□

Controlling the GIMIX Home

Most homes are wired by running 110 volts from the light being switched to a wall switch and then to the circuit breaker or fuse panel.

However, if you want two switches to control the same light or wall outlet then you must run an extra 110 volt wire in conduit from each switch. In addition you must use a special wall switch.

If you want to control the same circuit from more than two places, the cost becomes prohibitive. In addition, very few electricians know right off the top of their heads how to wire a circuit requiring control from more than two points. (Figure 1)

Relay control becomes desirable at the two switch control level, and they become necessary when you want control from more than two places.

A relay can be controlled from any number of switches. Wire the light to the relay and the relay to the panel using 110 volt wire, in conduit as necessary. The control wiring to activate the relay is 24 volts, and uses ordinary bell wire. Low voltage wiring is not required to be in conduit. Therefore, the cost of installing a single circuit can actually be less using the relay, when the cost of conduit and labor are also considered. (Figure 2)

The G.E. RR8 relay draws power only at the time of switching to either an "on" or an "off" state. It then stays mechanically latched at the ON or OFF position, until it is positively commanded to go to the other state. If the relay is on it will stay on until it receives an off signal, and any further ON signals will not cause anything to happen. Each relay can control up to 20 amps and up to 277 volts. The G.E. RR8 relay is U.L. approved and they have been in use for over 20 years. They are available through electrical supply houses and cost around $12 each.

Since all power for lights and other electrical items must be connected back to the breaker panel, the most convenient location for the relays is next to the breaker panel. Since the relays have four screw based terminals — two on the line side (breaker panel) and two on the load side (light or wall outlet), they are convenient for terminating several lights to each breaker. Only one wire is needed from the breaker to one of the relays on that breaker. The power can be run in a daisy chain fashion from relay to relay to feed the power from the breaker. (Figure 3)

To dim the lights we suggest using an extra relay in series with the light's relay. Place a diode, or a rotary

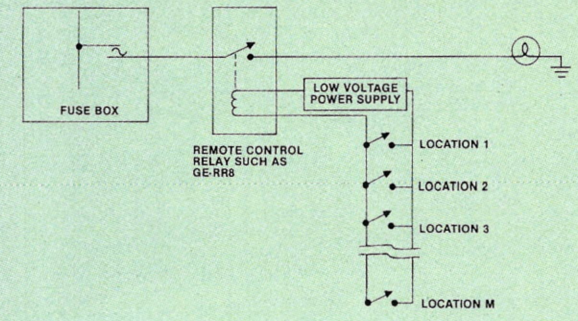


Figure 1.

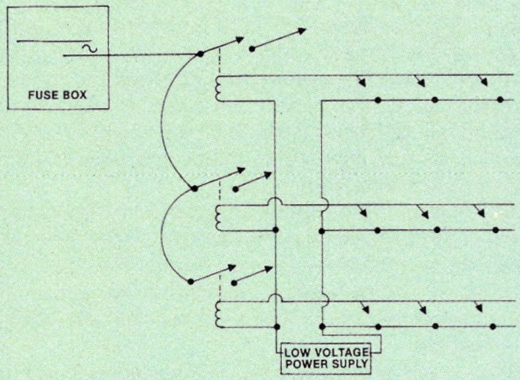


Figure 2.

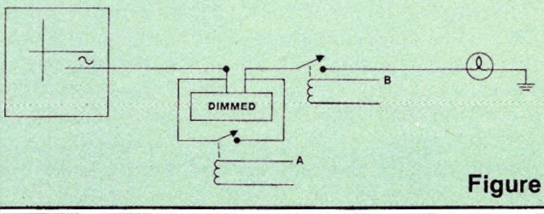


Figure 3.

RELAYS BA ON — FULL
RELAY B ON ONLY — DIMMED
B ACTS AS ON/OFF SWITCH

Figure 4.

dimmer if preferred, across the line and the load side. When the relay is off, current flows through the diode creating a dim condition. When the relay is on, the diode or dimmer is shunted, and the current path is unimpeded. (Figure 4)

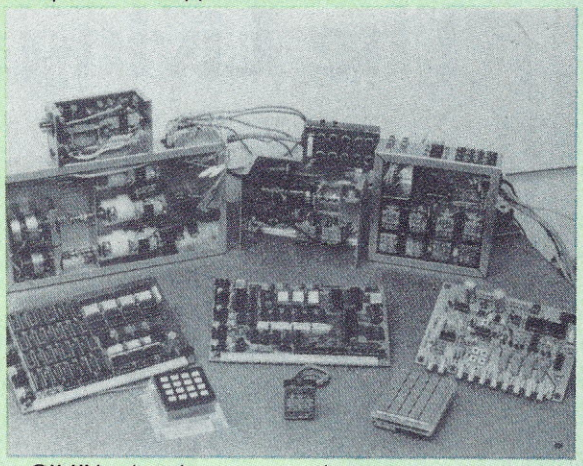
The low voltage side of the G.E. relay has four wires: the blue is the common, and the red wire is used to switch the relay on, while the black wire is used to switch the relay off. The fourth wire, yellow, is used for a pilot light to indicate remotely the status of the relay.

The GIMIX relay driver board terminal strip has connections for all four leads, plus it provides terminal tie points for wiring ON and OFF switches to each relay as desired. When switches are wired directly to the board, you then have a normal low voltage switching system, providing a "fail safe" method of controlling your lights, even when the computer is down or powered off.

There is no limit to the number of switches that can be directly wired to each relay, as all manually controlled switches are wired in parallel with each other. Normally open, push to close, pushbutton switches must be used as the manually operated remote control switches.

The GIMIX relay control board has several unique features, one of which is that it only requires two pairs of wires to communicate with the computer. One pair is used for communication from the computer to the relay driver board, which is generally located near the circuit breaker panel. The second pair of wires is for communications from the relay driver board back to the computer. These messages can either be relay status messages (relay 1 on, relay 2 on, relay 3 off, etc.) or error messages (last command had a parity error, relay won't respond, wrong board or relay responding, etc.). The relay driver board is an intelligent device and can be commanded to constantly scan all attached relays (it does this by interrogating the blue wire to the G.E. RR8 relay).

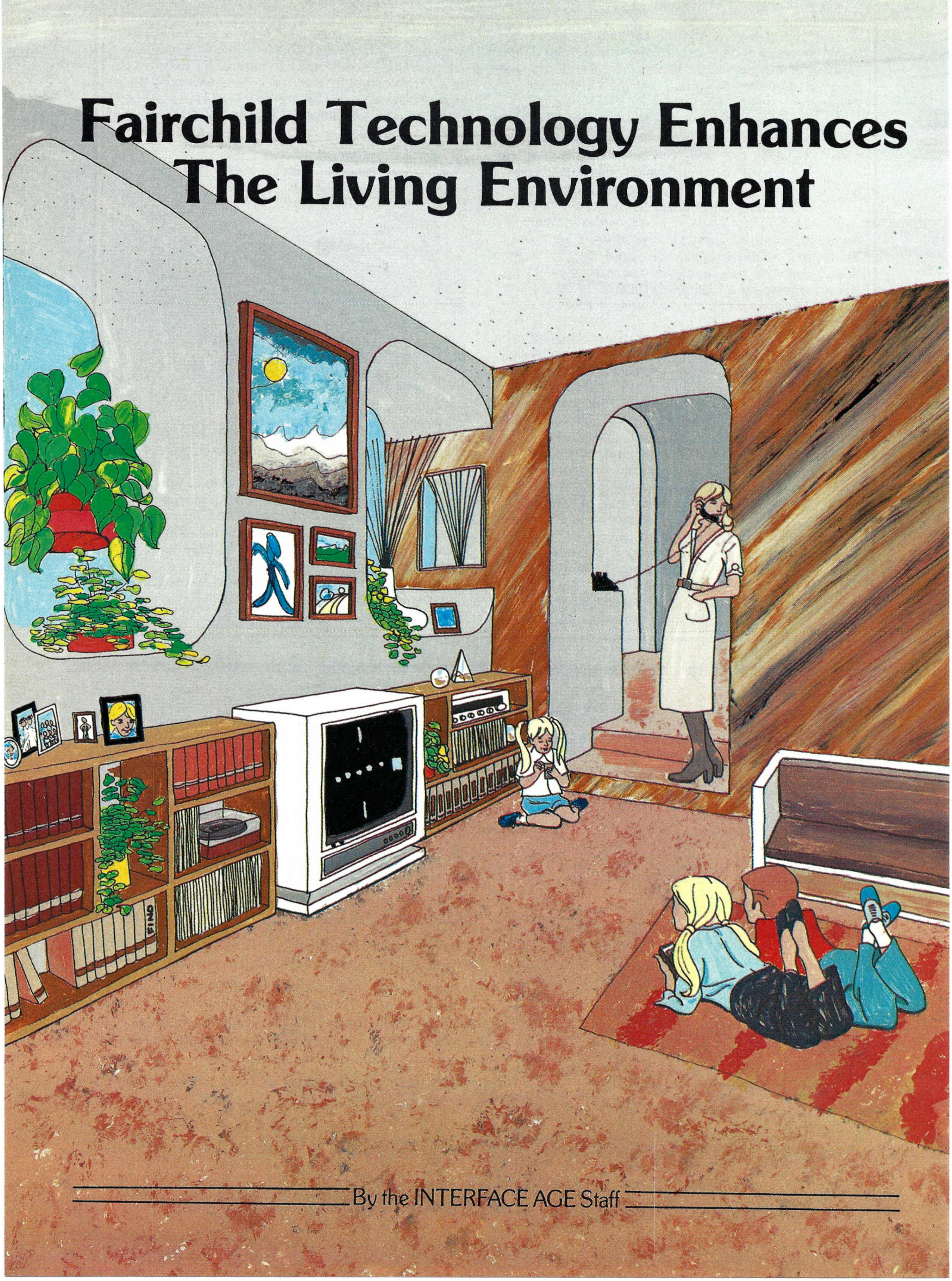
Each relay driver board can control 31 20-amp circuits. Each computer port can control four relay driver boards, for a total of 124 controllable circuits per port. Obviously the only limit is the speed of the computer and the number of serial current loop I/O ports the computer will support.



GIMIX also has a complementary opto-coupler board, which will accept 34 ON/OFF input circuits. In addition the corporation also markets a touch-tone receiver circuit, used to convert touch-tone signals into 8-bit ASCII characters for processing by any CPU. Both the opto-coupler board and the 2-wire tone receiver circuit interface to the CPU through an 8-bit parallel input port.

Application notes and examples of home automation are available from GIMIX Inc., 1337 W. 37th Place, Chicago, IL 60609, (312) 927-5510.☐

Fairchild Technology Enhances The Living Environment



By the INTERFACE AGE Staff

A hundred years ago our ancestors were living in homes heated by wood burning fireplaces and lighted by kerosene lamps or candles. They cooked over wood burning stoves and communicated with friends and relatives through the mail or by messenger. Home entertainment revolved around good books and lively conversation, or in some cases, the family piano.

Times have changed. The 1970s have revolutionized the American home. Electricity is king, and it has given birth to the 70s phenomenon of the "total electric home." Through the magic of a little thermostat on the wall, we can heat our homes in the winter and cool them in the summer. We cook our meals in an electric oven or we can zap a meal in minutes with a handy microwave. Iceboxes can only be found in antique stores these days; our modern refrigerators and freezers can store enough food for a year, make an endless supply of ice cubes, keep themselves frost free and generally do everything but walk the dog in the morning.

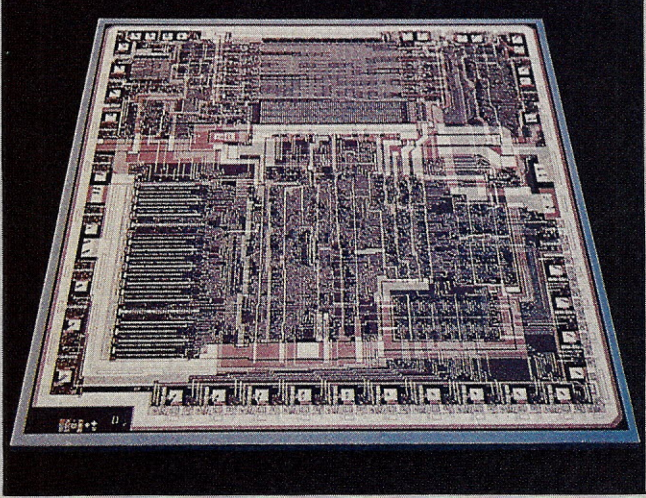
Communication has taken on a whole different meaning as well. The revolutionary little device Alexander Graham Bell gave us a century ago has been the breakthrough in the communications barriers. Today, we can pick up the kitchen phone in Kansas and be casually chatting with a friend in Japan within seconds.

Through the wonder of the phonograph (there's an outdated word) we're able to listen to Beethoven, Bach or the Rolling Stones without ever leaving the comfort of our living rooms. World events are brought to us on television as they are happening.

The list goes on and on. Technology has brought us an incredible distance in the span of a century. It will take us even further during the next decade.

The computer society we live in today has been recently chronicled by experts and observers alike. The phenomenon of electricity is being overshadowed by its offspring, the computer. As little as 15 years ago, computers were only found in high technology businesses and in futuristic fantasies such as the film "2001." Today they are commonplace; we have computerized cash registers in fast food markets, kids have computerized calculators to help them through math and an increasing number of Americans have computers in the home just for fun.

The 9440 Microprocessor

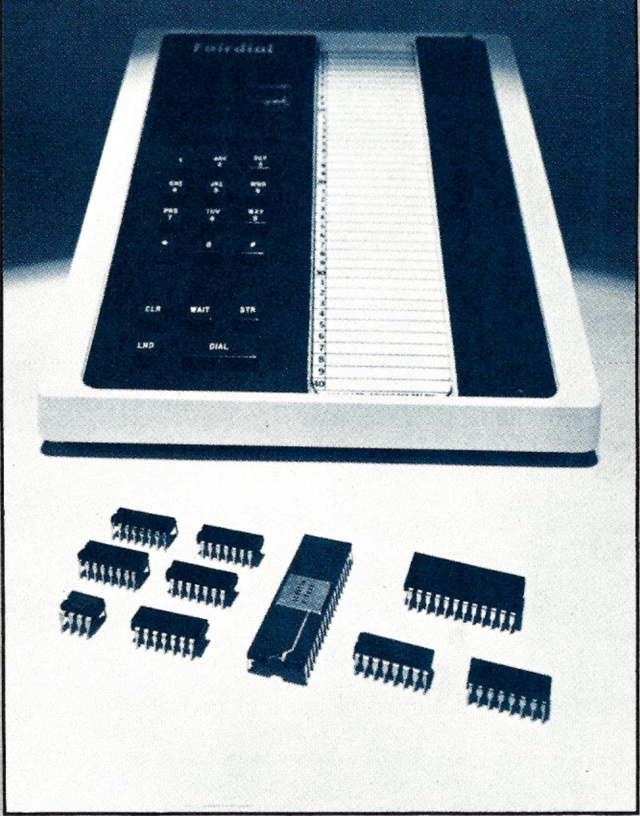


Photos courtesy of Fairchild Camera and Instrument Corporation

With this rising popularity comes an increased competition among manufacturers to produce affordable computer products for home use. A consistent leader in this field has been Fairchild Camera and Instrument Corporation of Mountain View, California. With its 9440 Microflame™ microprocessor, which allows a programmer to execute a minicomputer instruction set with minicomputer performance; the Fairchild SH1549 hybrid tuner control/memory for multi-station AM, FM and TV tuners; the Channel F Video Entertainment System; and more recently, the Fairdial™ telephone repertory dialer, the company has made its name something of a household word in the home computer arena.

The Fairdial component set has given Ma Bell a run for her money since its introduction in the latter part of 1978. It is a kit of 10 semiconductor components that can be used to construct a telephone repertory dialer with the capability to store up to 44 telephone numbers of 12-digit length.

The Fairdial Telephone Dialer



The kit is based on an Isoplaner NMOS telephone controller circuit designated as the CET 200. It includes two MOS 1K RAMs, two CMOS gate circuits, and five TTL circuits in addition to the controller circuit.

In a prototype dialer designed by Fairchild to demonstrate the capabilities of the component set, 40 numbers are stored in a "telephone book" memory, two are stored in scratch pad memory, one is stored in a "last number dialed" memory and one is stored in an entry buffer memory.

The component set can be interfaced with standard 7-segment LED displays to display the number being entered or dialed. Telephone numbers are automatically formatted into groupings of area code, prefix and station number.

One of Fairdial's biggest appeals is its offer of pushbutton dialing economy with pulse type signaling. The Bell System Touch Tone offers the convenience of pushbutton dialing, but the feature is not available in all parts of the country.

Incorrect dialing has long been a frustration associated with the standard rotary dial. Miss one digit and you have to begin the slow dialing process all over again. With Fairdial, dialing the correct number is speeded up several ways. First, entering the telephone number is a separate process from dialing the number. Although the entire number is entered through the keyboard number keys, it is not transmitted to the telephone line interface until the DIAL key is pressed. If the caller enters an incorrect digit, he need only press the CLR key to re-enter the number.

With the Fairdial it is also possible to call a number a second time without having to re-enter it. The last number dialed is automatically stored in a separate memory register until another is entered. If the number dialed is busy or is not answered, the caller can redial it at any time by pressing the LND (last number dialed) key. He can verify the number by pressing the LND while the telephone is on the hook. The number appears on the display but dialing doesn't take place until the receiver is lifted.

The Fairdial also contains a telephone-book memory for a collection of those numbers the individual or family most frequently calls. The memory, which has a 40-number capacity, is configured into two pages of 20 12-digit numbers. The locations are associated with a specific control button so that when the receiver is off the hook and the associated button is pressed, the number stored in the memory is copied into the display and dialed. The caller differentiates the two pages by a page-select function, which is activated either by a separate control switch or a two-position pushbutton. The numbers loaded in the memory remain until they are placed or cleared.

The Fairdial is compatible with telephone systems using either old or new pulsing systems. It is capable of signaling the central office at the standard 10-pulse-per-second rate or the newer 20-pulse-per-second rate currently available in some areas of the country.

For the sophisticated home computer buff, Fairchild offers the 9440 Microflame 16-bit 19MHz microprocessor. According to Dr. Thomas A. Longo, vice president and chief technical officer at Fairchild, the device is the industry's first to bring full minicomputer capabilities to the microprocessor world in a variety of applications.

The company offers an introductory kit consisting of the 9440, sixteen 4.096-bit memories, the SSI/MSI components required for memory control, along with introductory software and manuals. The kit enables users to construct a working system at the board level in their own board format.

The 9440 Microflame is a complete minicomputer CPU on one chip, packaged in a 40-pin DIP, and can execute the Data General NOVA 1200 instruction set. It is based on Fairchild's Isoplaner integrated injection logic (I^3L)™ technology and is designed to operate with power supply and performance-compatible TTL dynamic memories. The TTL dynamic memories also are fabricated with an I^3L technology.

Data and instructions are stored in external memory and a 16-bit wide three-state information bus carries both data and addresses between the CPU and other computer circuits.

Intrinsic memory capacity of the 9440 is 32,768 16-bit words, and the I/O ports can serve up to 63 peripheral devices using programmed I/O, interrupt-driver I/O or direct memory access.

According to Dr. Longo, the I^3L technology used in the 9440 Microflame TTL dynamic memories combines Fairchild's Isoplaner dielectric isolation techniques with integrated injection logic circuitry, enhancing both speed and density. These devices can achieve switching speeds comparable to low power Schottky TTL devices and packing density up to 250 gates per square millimeter.

Just when the old folks were getting used to television and its usurping of their old radio programs, along came TV video games. This latest technological handiwork was originally intended for the youngsters in the household, but it soon became apparent that the kids would have to edge their parents out of the way before they could play.

Fairchild is in the forefront of the video game market with the Channel F™ Fairchild Video Entertainment System. Seventeen Videocart™ interactive game cartridges, offering 30 games with over 500 variations, are now available in this system. For the masters of war, there are combat games such as Desert Fox; for the sports addict, baseball; and for the more intellectually oriented, Memory Match.

The components for the system include a game console, a pair of eight-way hand controls, an adaptor box and the optional game cartridges. The adaptor serves as an interface permitting a player to switch from normal television viewing to video game and back again.

Incorrect dialing has long been a frustration. Miss one digit and you have to begin again.

The heart of the Fairchild system is an F8™ microprocessor. Both the microprocessor and its associated 16K RAM are incorporated into the game console, making this system the first truly programmable game center.

The instructions for the various games available are retained in ROM program stores. Instructions for hockey and tennis are contained in ROMs within the console while program stores for other games are mounted on printed circuit boards within the Videocart cartridges.

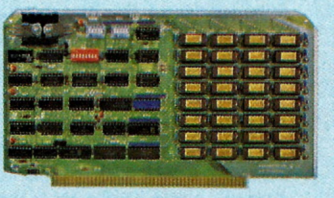
Data in the ROMs establishes operating parameters for the microprocessor and loads preliminary data into the RAM. Additional transient data are supplied by the five selector control buttons on the console and by the hand controls. The games are produced on the TV screen through interaction of the ROM, RAM, microprocessor and controls.

The player has the opportunity to select one of four difficulty modes, one of four time limits and with some cartridges, variations of a particular game. The system allows players to "freeze" the game action for an indefinite period of time, then resume play where they left off. After the action has been frozen, the Channel F system can be reprogrammed for a new time limit, difficulty mode or both.

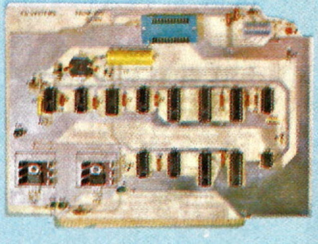
The Fairchild video game system can be used with any make or size of television set, either black and white or color.□

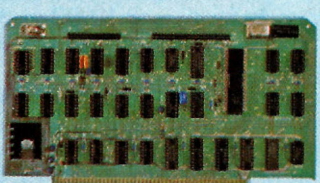
4 New Boards & the SDS-200 join State-of-the-Art SD Technology

Versafloppy II–Dual Sided, Dual Density, Single Density, Single Sided, 5-inch and 8-inch flexible drives. Controls up to four drives simultaneously, any and all combinations. Extremely powerful SDOS operating system.

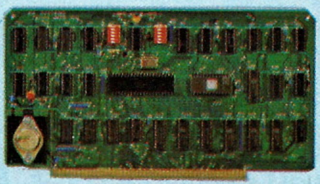


MPB-100–Z80 Central Processing Unit. S-100 Bus Compatible. Front panel useage optional. 2MHz or 4MHz operation speeds. Power-on jump to 4K boundaries. On board Sockets for up to 2K PROM.





ExpandoRAM II – Expandable from 16K to 256K Bytes. Page Mode operation allows up to 4 boards in use simultaneously on the same S-100 Bus for multiple users. Compatible with 8080, 8085 and Z80 CPU's.



PROM-100–Programming board for PROM/EPROM (2708 • Intel 2758, 2716, 2732 • T.I. 2516) 25VDC Programming Pulse generated on-board. Software provides for reading of object file from disk or PROM.

The New Expandable Capacity SDS-200

Features: On-line RAM from 64K to 256K (Expandable) • Over 2,000,000 bytes of on-line diskette storage (expandable to over 4,000,000 bytes) • Over 8,000 bytes of available PROM • 12-inch Video Screen (24 lines of 80 characters) • Z80 CPU • Parallel and RS-232 Serial Input/Output Ports • Full upper and lower case keyboard • Numeric pad and cursor controls • 256 characters available on video display • Blinking, Underlining, Reverse and protected field enhancements • Key-lock power switch • Power-on bootstrap • Programmable Baud Rates • Addressable cursor • Compatible with C-Basic, E-Basic, Cobol, Disk Fortran, Microsoft Disk Basic, SDOS, CP/M and hundreds of compatible software packages. Operation speeds to 4-Megahertz • flicker free video display.

For product information & the location of your local SD Systems Dealer: Call or Write P.O. Box 28810, Dallas, Texas 75228. 800-527-2304; 214-271-4667; TELEX 73-0151.

State-of-the-Art Computer Boards

VDB-8024 Video Display Board The on-board Z80 gives a new dimension to video technology. Utilizes on-board RAM and ROM. Assembled & Tested or Kit versions are available.

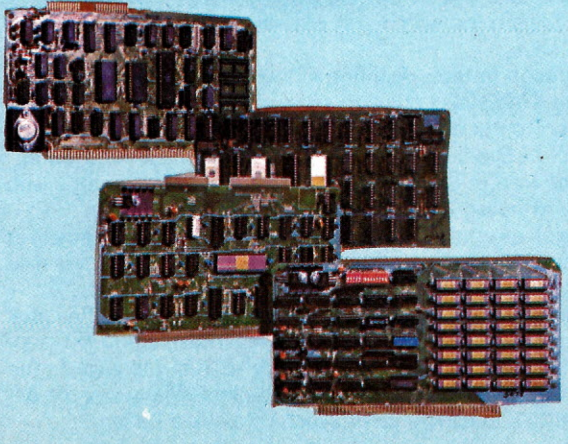
SBC-100 Single Board Computer Z80 Microprocessor, RAM, PROM, 4 channel Counter/Timer, S-100 Bus Compatible, Serial and Parallel I/O's.

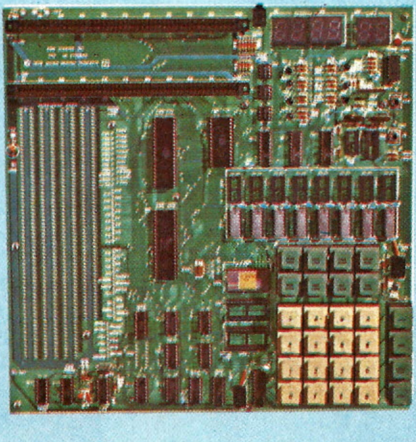
Versafloppy Flexible Disc Drive Controller for up to four drives simultaneously: Mini's, full size, one or two sided, single density. IBM 3740 Compatible.

ExpandoRAM expands from 8K to 64K on the same board. Switch selectable banks & Write Protect.

The SDS-200 is the technological leader of microcomputing. Flexibility and expandability give you a computer that will grow with you whether your application is a total business system, scientific analysis or process control. Operator ease and operator interaction have been designed into the SDS-200. Software compatibility is unprecedented. The Random Access Memory capacity and operation speeds allow the use of high level languages. The SDOS operating system is the most powerful in the industry giving unheralded flexibility and compatibility.

The SD Systems Business Software packages interact with the operator to give you complete accounting control. The features of the SD Business Software provide for multi-level and multi-company accounting. Clerical costs are reduced, work improvement is realized through orderliness, accuracy and consistency, management supervision is reduced and management reports are provided on a timely basis.

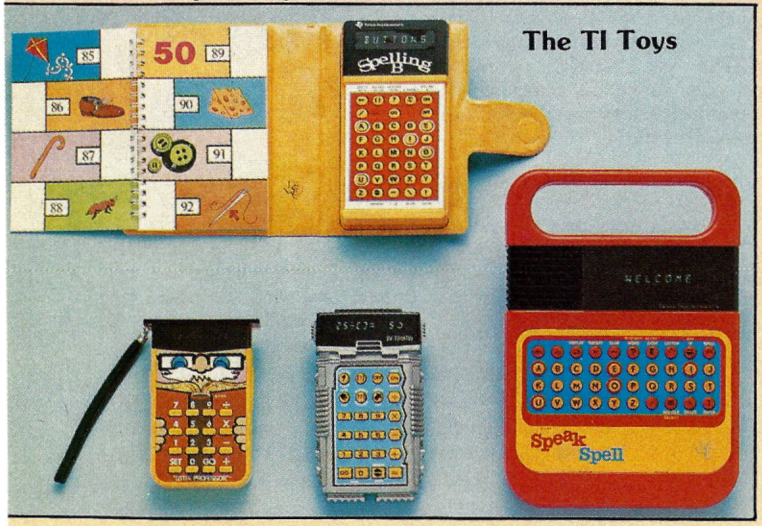




The **Z80 Starter** Kit by SD Systems uses the powerful Z80 microprocessor as the heart of the complete microcomputer on a single board. Learn a step-by-step introduction to microcomputers with a keyboard and display, audio cassette interface, PROM programmer, wire-wrap expansion area, 4-channel Counter Timer and on Board RAM and PROM. Complete Operation and Instruction Manual included. ZBUG Monitor in ROM

Electronic Games
The First Step to Home Computers

By Terry Costlow, Assistant Editor



The TI Toys

Until recently, the word *computer* brought thoughts of a hard to use machine that would take away jobs and dehumanize society. During the last few years many things have changed, and home systems have begun making the predicted inroads towards "a computer in every home."

While fears about computers have not completely dissipated, sales of home units continue to increase. Many people have learned that computers can save time by doing things more efficiently. Adults have learned to cope with the new technology.

For the younger set, computers are just another fact of life. And as children bring computerized toys into the home and adults discover that home computers are not as difficult to use as they once were, the systems will enter the home more rapidly during the near future.

Two of the leaders in producing equipment for large consumer markets are Texas Instruments and Mattel Electronics, a division of the huge toy manufacturer.

Mattel's Updated Football

Mattel Electronics has already jumped form the realm of computerized toys into preliminary production of a home system. Texas Instruments is currently staying with its line of toys, calculators and the like, although officials don't deny persistent rumors that the company is working on a home system.

TI has become a mainstay in the field of designing educational games for children. Beginning with the mathematical challenger, Little Professor, in 1976, TI has introduced several games for elementary and pre-school children.

As are most of the company's toys, the Little Professor is based on the TMS1000 four-bit microcomputer. The four-ounce, hand-held unit is programmed with over 1,600 math problems, which are broken into four levels of difficulty so the child won't outgrow it rapidly.

During the 1976 Christmas season, the Little Professor was almost alone as a computerized toy. But after it proved fairly successful, other companies joined in. During the 1977 holiday season, there were about 12 computerized toys on the market. TI's entries included another math-oriented game, DataMan. Their novelty brought high sales.

"The initial toys went over like gangbusters. Everyone saw how well they did and next year some of the major toy manufacturers got into the act," says Jim Mullen, director of technical communications for Texas Instruments.

The 1978 Christmas season made history. Shoppers couldn't pick up a major magazine or newspaper without seeing either an ad or a story about the "new" type of toys. And the toys sold so rapidly that reading about them was as close as many people every came to having one.

TI's Spelling Bee

But while the traditional toy manufacturers designed games that were geared mainly towards entertainment for either adults or children, TI stuck with educational games for younger children, including the popular talking Speak 'n' Spell. And they don't plan to change that strategy.

"We plan to continue with educational toys because they make the best use of our existing resources. We're geared mainly toward the watch, calculators and specialty devices," Mullen says.

In addition to their own line of toys, the Dallas corporation is selling TMS1000 chips to several toy manufacturers. Coleco is currently designing an astrological toy that uses the chip. Milton Bradley uses it in a computer-controlled dump truck and Parker Brothers is combining chip technology with a standard board game format for a detective game.

A major question for those designing home computers is "Will the success of computerized toys translate into increased interest and acceptance for full-blown home systems?"

"Absolutely," Mullen says enthusiastically. "In that regard, it behooves the big companies to explain to the people how these machines operate. We're trying to explain the technology of the machines to the buyers. Many people believe that it's electrical just because it uses a battery.

"We're trying to explain that it isn't just an electrical device. It's a programmable computer, and the same type of chip is used in the home computers. The only difference is in the software. It's all in what you program the computer to do."

Mattel Electronics officials also feel that toys will pave the way for the futuristic ideal of a computer in every home. They feel that the success of their games will help people overcome their thoughts that Mattel is a toy company, not a serious electronic organization.

Mattel joined the computer revolution with hand-held toys, including football and a TV-takeoff, Battlestar Galactica. Those two toys were big hits during the computer Christmas last year. With an update of football and a new baseball game being added to the lineup, Mattel again hopes to score heavily in the realm of technological toys.

They also hope to parlay their acceptance as a computer-oriented company into further success with a new home system, Intellivision. Although the decision to design a home computer system was made in 1976, the success of the hand-held games did nothing to hold the designers back.

The resulting hardware modules and software cartridges will be marketed with a multi-million dollar advertising campaign this fall, just in time for Christmas. The unit, which is made of two separate modules selling for about $250 each, connects to a standard television set. The modular concept is designed to keep the cost acceptable.

"We don't think that most families are ready for a $500, one-shot investment, so we're offering them the chance to get started with just one component and then expand when they think they can," says Jeffrey Rochlis, president of Mattel Electronics.

The modules include a master controller which is designed mainly for entertainment and a 64-key keyboard component that can be used for more standard uses of home computers.

During the designing of the computer, Mattel officials made full use of the company's huge research and development department to find out what the "average consumer" thought a home computer should do. Intellivision was created to reach this mass audience.

TI's First Watch

"The computers that are on the market now are, from the consumer's point of view, solutions looking for a problem. We are a consumer-oriented company. We looked at those people's needs and geared our product towards them," Rochlis said.

To avoid the problems of teaching people how to program the computer, Mattel put all the game software on ROM cartridges and others on cassettes.

"Our system may not be accepted by the computer aficionado. We deliberately eliminated the programming so it would be simple enough for any member of the family to use. The consumer doesn't want the burden of learning a language or the burden of learning to program. To them, the systems that are available now require a disproportionate amount of learning time," Rochlis says.

Mattel's Baseball

Some of the software Mattel has put on cassette includes income tax prepartion, menu and exercise planning, and foreign language studies.

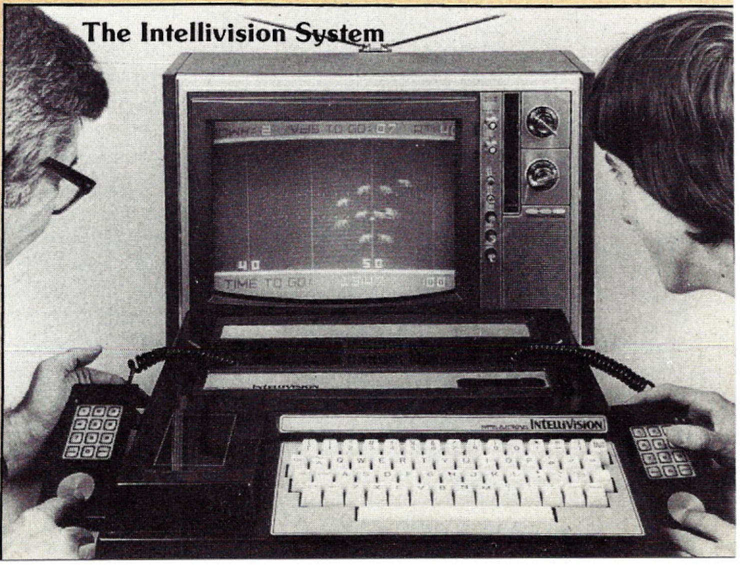
The foreign language tapes utilize four tracks on the tape. One on each side holds the programming, while the other two tracks are for verbal recordings. The tape allows the user to record his or her voice to see how their pronunciation compares to the computer's.

By designing the software to appeal to all members of the family from preschoolers to both parents, the company planned to help make the computer enjoyable to all members of the family to enable the purchaser to justify his expenditure.

After the initial sales at Christmas, Mattel plans to send out flyers detailing new software developments to all who send in warranty cards.

Rochlis sees home computers coming of age during the eighties, but not without an ongoing educational process.

"It will take several years of use before the computer is considered to be a standard household appliance. But as we looked at consumer electronics, we saw a large market for computers that would continue to grow. I can see one day in the future when hardware costs only about $100 and is a utility that's installed like a telephone."☐



The Intellivision System

Home Applications for the 6800

By David E. Shambaugh

This article deals with a Motorola MEK 6800 D2 kit, which sells for about $225. The "D2" kit comes with several capabilities: an MC6800 MPU; three MCM6810 RAMs (128x8 each); two MC6820 Peripheral Interface Adapters (PIA); one Asynchronous Communication Interface Adapter (ACIA); one MC6871 Clock Generator and one MCM6830 ROM with JBUG monitor.

When assembled, the kit consists of these parts separated on two boards, the MPU board and the display board. The display board contains the six 7-segment displays, the keyboard and the audio cassette circuitry. This allows the ACIA and second PIA available to the user in the event that the user has access to an RS-232 TTY terminal. Wire wrap space is available for up to twenty 16-pin DIP packages for user designed circuitry.

The kit as built with the JBUG ROM, is a very powerful device which can be used for all sorts of applications around the home.

This article tells you how to take a D2 kit and make:

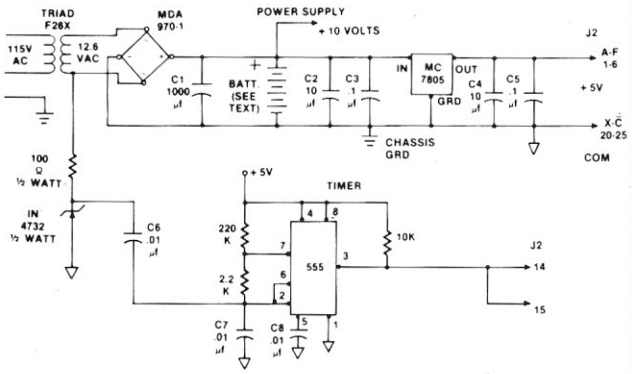
A Digital Clock. Hours, minutes, and seconds will be continuously displayed on the six 7-segment displays. The circuit in Figure 1 is a timer which replaces the 60 cycle line frequency during power failures. Thus the clock keeps running when the power is out, assuming that the battery back up system shown in Figure 2 is added. Another feature is that by changing data in two memory locations, as shown in the program listing, the 12-hour clock can become a military 24-hour clock.

A Burglar Alarm. This alarm, wired as shown in Figure 3, is unique in that is is an "adaptive" alarm system. This is done by having the MPU look at the state of the alarm switches and "memorize" each one. The MPU then comes back a few seconds later and looks at all of the switches again. If any switch(es) has changed, the alarm is triggered for an adjustable period of time. At the end of this time the MPU shuts off the alarm and looks at the state of the switches again. This state is now accepted as the "new" condition and the entire cycle starts over. With this system a switch can fail, be destroyed, be tripped or whatever and the rest of the alarm is not disabled.

A Timer/Control. The wiring for the timer output is shown in Figure 4. Each time the clock program updates the hour count, it checks to see if the program wants the timer output turned on. If it does, it turns the output on, then checks every minute count to see if it has been on for the desired time. If the time is up, the output is shut off. This output will control a sprinkler system or other home device.

HARDWARE AND SUPPORTING SOFTWARE

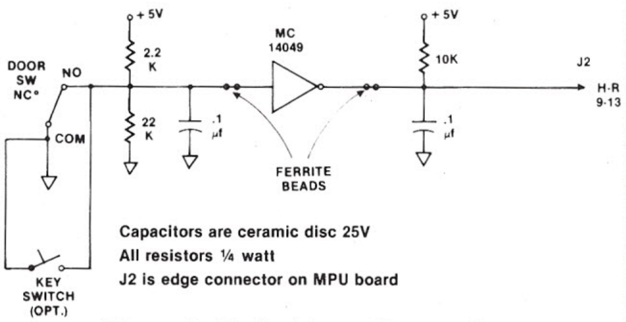
As stated, Figure 1 shows the interrupt timer and power supply. The power supply uses four each Gates rechargeable lead acid cells at 2 volts each. This 8-volt source is regulated down to +5 volts with a 3-terminal voltage regulator (MC 7805). The batteries are supplied with a charge as long as 60 Hz power is present by using a 12.6 volt, 2 amp transformer and full wave bridge rectifier with a 1000 µf capacitor. This produces 10V DC, with battery loading the circuit and is within specifications supplied by battery manufacturer.



C1,2,4 Electrolytic 25V
C3,5,6,7,8 Ceramic Disc 25V
All resistors ¼ watt except where noted
J2 is edge connector on MPU board

Figure 1. Power Supply and Interrupt Timer.

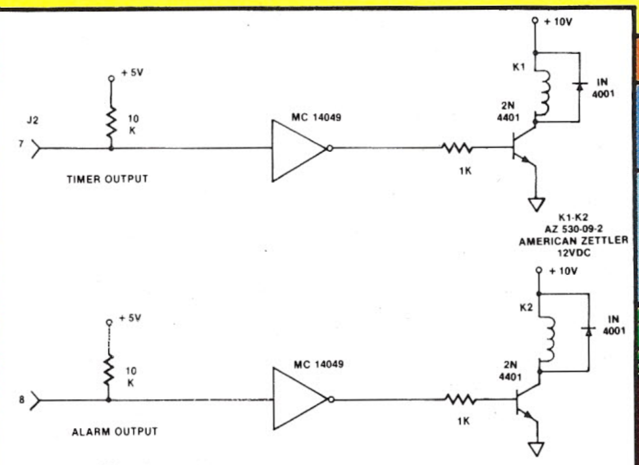
The interrupt timer (also in Figure 1) uses a 555 timer wired in the astable mode. The oscillation rate is slightly faster than 60 Hz (the period is approximately 17 msec and the pulse is approximately 150 µsec), which is more than long enough for recognition by the MPU as a valid interrupt.



Capacitors are ceramic disc 25V
All resistors ¼ watt
J2 is edge connector on MPU board

Figure 2. Typical Input Connection.

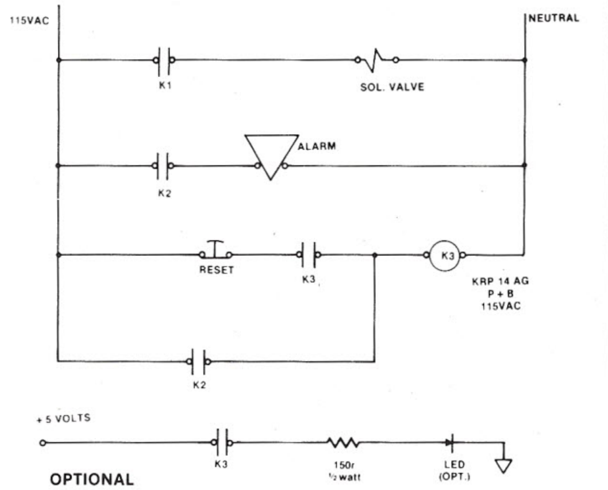
Figure 2 shows a typical input connection. An MC 14049 inverter buffer is used to protect the PIA from noise and voltage spikes on the input lines because of the length of the wire going to the switches. There are also numerous pull ups, pull downs and bypass capacitors for elimination of false triggers. There are also other ways of designing around noise problems. Ferrite beads are used in some places to get rid of the interference caused by CB enthusiasts and their 1 KW linears. The switch connection is shown, and can be wired in the Normally Open or Normally Closed configuration. It is shown wired Normally Closed in this figure because it also shows the use of a Key Switch which is used to short the switch for entering the premises without tripping the alarm.



All resistors ¼ watt
J2 is edge connector on MPU board

Figure 3. Typical Output Connection.

Figure 3 shows a typical output connection. The outputs from the PIA are also protected by an MC 14049 inverter buffer as on the inputs. The outputs are not as critical for noise problems as the inputs, so are much simpler.



OPTIONAL
Key Switch & LED are mounted together on switch panel.

Figure 4. Optional Security Indicator.

Shown in Figure 4 is a latching relay circuit used in the system for a special feature. The relay latches through a Normally Closed reset push button every time the alarm is sounded. When the relay is energized, it lights a solid state LED mounted on a switch panel with a key switch by the door used for entering.

This is important in that it will tell if the alarm system was triggered while no one was in. If the LED is on, be cautious in entering the premises in case an intruder is still present.

Now on to some explanations on the supporting software. The main program, which is the digital clock, works in this way. It first looks for an interrupt which is present out at PIA location HEX 8006. When this input goes high it checks the validity by also reading the MSB of the input at HEX 8006. This bit is also high when the interrupt occurs. When the MSB is high it means a minus number is there. This is accepted as a valid interrupt and it updates the SSEC accumulator which keeps track of the 1/60 second count. If the data is not a minus number, it goes back and looks for another interrupt. This is a noise eliminating check. The program counts sixty interrupts and updates the seconds accumulator. It then jumps to the display subroutine in the JBUG ROM and updates the display. Every time it counts sixty seconds, it updates the minute accumulator and jumps to the display routine in the JBUG ROM again and updates the appropriate display. This happens with the hour accumulator in the same way at every hour.

Every time the display is updated the program jumps to the timer subroutine starting at HEX 009B and checks to see if the hour accumulator, HEX location 0003 and the Hour Setpoint, HEX location 0009 are the same. If they are the same, it compares the Minutes Accumulator, HEX 0004 and the Minutes Setpoint, HEX 000A. If they are the same, it sets the timer output high and also sets the Timer Triggered Flag, HEX 0007, and returns to the main program. Once the Timer Triggered Flag is set, the program compares the Minutes Accumulator, HEX 0008, with Delay 1, HEX 000B, and when they are the same will turn the timer output back low again and clear the Timer Triggered Flag.

Every time the program updates the second count it jumps to the Alarm Subroutine at location HEX 00CE. In this routine, it compares the PIA locations at HEX 8004 and 8006 to the two Alarm Status Registers at HEX 000F and 0010. In the first few steps of the main program the data at these two PIA locations was loaded into the two Alarm Status Registers. When the Alarm Subroutine compares the PIA location with the appropriate Alarm Status Register, it should see no difference unless a switch condition was changed. If a difference occurs in either place, the delay register is incremented and compared to the delay which was input at location HEX 00DE.

Once this seconds count is finished and the condition is still present, the alarm output is set high and the Flag 1 register, HEX 000E is set. This delay eliminates noise and false triggers. Once the flag is set, the program counts seconds in the Delay 3 register, HEX 000D and compares them with the delay put in at location HEX 0102. This the alarm on time. Once this delay is counted out, the alarm output is turned back low and the PIAs (HEX 8004 and 8006) are read and loaded into the two Alarm Status registers. This puts the new data in the registers for future comparisons. This means if a door or window is opened, the alarm sounds for an adjustable time and scares the intruder off. When the alarm is reset, the window or door left opened is taken as a normal condition. The alarm will sound off again if another door or window is opened or if the opened door or window is closed. A HEX entry of 01 at 0102 will disable the alarm.☐

Location	Load
0003	Present Hour
0004	Present Hour
0005	00
0006	00
0007	00
0008	00
0009	Hour you want timer on (00-23)
000A	Minute you want timer on (00-59)
000B	Minutes you want timer to be on (00-FF)
000C	00
000D	00
000E	00
000F	00
0010	00
007F	12 for 12 hour clock — 24 for 24 hour clock
0083	01 for 12 hour clock — 00 for 24 hour clock
00DE	01 for 1 second delay[1]
0102	Alarm on time (00-FF)[2]
	01 disables alarm

[1]Any HEX number from HEX 00-FF can be put in here, but 01 is the best number for noise elimination and fast recognition of intruder and will allow alarm to trigger after 1 second.

[2]Any HEX number from HEX 00-FF can be put in here also. HEX 00 gives you 256 seconds of alarm time. HEX 01 will disable alarm. Any other HEX number will provide the appropriate alarm on time.

Program Listing Follows

PROGRAM LISTING

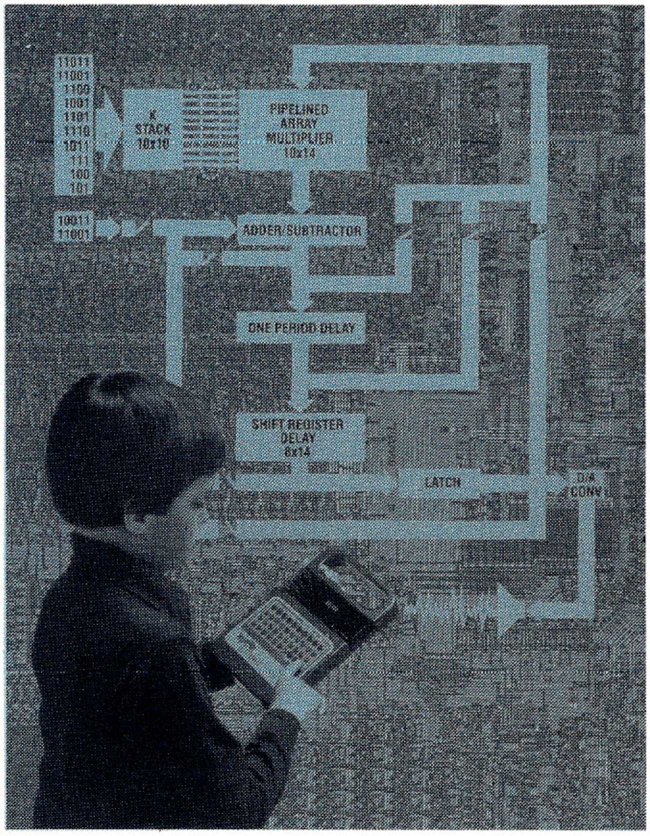
```
                              NAM    DAVE
                              OPT    P
          8004      PIADA     EQU    $8004
          8005      PIACA     EQU    $8005
          8006      PIADB     EQU    $8006
          8007      PIACB     EQU    $8007
          E0FE      OUTDS     EQU    $E0FE
          E275      MISD01    EQU    $E275
          E327      REGST6    EQU    $E327
          E31C      REGST5    EQU    $E31C
          A000      INTLOC    EQU    $A000
    0000                      ORG    $0000
    0000  7E 0011             JMP    START
    0003  0001      HR        RMB    1
    0004  0001      MIN       RMB    1
    0005  0001      SEC       RMB    1
    0006  0001      SSEC      RMB    1
    0007  0001      TIMTR     RMB    1
    0008  0001      MINAC     RMB    1
    0009  0001      HRSPT     RMB    1
    000A  0001      MINSPT    RMB    1
    000B  0001      DEL1      RMB    1
    000C  0001      DEL2      RMB    1
    000D  0001      DEL3      RMB    1
    000E  0001      FLG1      RMB    1
    000F  0001      AS1       RMB    1
    0010  0001      AS2       RMB    1
    0011  01        START     NOP
    0012  0F                  SEI
    0013  4F                  CLR A
    0014  97 06               STA A  SSEC
    0016  97 07               STA A  TIMTR
    0018  B7 8007             STA A  PIACB
    001B  B7 8005             STA A  PIACA
    001E  97 0C               STA A  DEL2
    0020  97 0D               STA A  DEL3
    0022  97 0E               STA A  FLG1
    0024  CE 03FF             LDX    #$03FF
    0027  FF 8006             STX    PIADB

    0084  97 03               STA A  HR
    0086  96 05     SETDSP    LDA A  SEC
    0088  BD E275             JSR    MISD01
    008B  96 04               LDA A  MIN
    008D  BD E327             JSR    REGST6
    0090  96 03               LDA A  HR
    0092  BD E31C             JSR    REGST5
    0095  20 04               BRA    TIMCK
    0097  8B 01     CKDEC     ADD A  #01
    0099  19                  DAA
    009A  39                  RTS
    009B  96 07     TIMCK     LDA A  TIMTR
    009D  2B 1E               BMI    X3
    009F  96 09               LDA A  HRSPT
    00A1  91 03               CMP A  HR
    00A3  27 02               BEQ    X1
    00A5  20 26               BRA    X5
    00A7  96 0A     X1        LDA A  MINSPT
    00A9  91 04               CMP A  MIN
    00AB  27 02               BEQ    X2
    00AD  20 1E               BRA    X5
    00AF  86 80     X2        LDA A  #$80
    00B1  97 07               STA A  TIMTR
    00B3  7F 0008             CLR    MINAC
    00B6  86 01               LDA A  #$01
    00B8  B7 8006             STA A  PIADB
    00BB  20 10               BRA    X5
    00BD  96 08     X3        LDA A  MINAC
    00BF  91 0B               CMP A  DEL1
    00C1  27 02               BEQ    X4
    00C3  20 08               BRA    X5
    00C5  86 00     X4        LDA A  #$00
    00C7  B7 8006             STA A  PIADB
    00CA  7F 0007             CLR    TIMTR
    00CD  3B        X5        RTI
    00CE  96 0E     ALM1      LDA A  FLG1
    00D0  26 24               BNE    X6
    00D2  B6 8004             LDA A  PIADA
    00D5  D6 0F               LDA B  AS1
```

```
00040 002A CE 0045            LDX     #INTRG
00041 002D FF A000            STX     INTLOC
00042 0030 CE 0004            LDX     #$0004
00043 0033 FF 8004            STX     PIADA
00044 0036 B6 8004            LDA A   PIADA
00045 0039 97 0F              STA A   AS1
00046 003B B6 8006            LDA A   PIADB
00047 003E 97 10              STA A   AS2
00048 0040 01                 NOP
00049 0041 0E                 CLI
00050 0042 7E E0FE            JMP     OUTDS
00051 0045 01        INTRG    NOP
00052 0046 B6 8006            LDA A   PIADB
00053 0049 2B 01              BMI     INT2
00054 004B 3B        INT1     RTI
00055 004C 7C 0006   INT2     INC     SSEC
00056 004F 96 06              LDA A   SSEC
00057 0051 81 3C              CMP A   #$3C
00058 0053 26 F6              BNE     INT1
00059 0055 7F 0006            CLR     SSEC
00060 0058 BD 00CE            JSR     ALM1
00061 005B 96 05              LDA A   SEC
00062 005D 8D 38              BSR     CKDEC
00063 005F 97 05              STA A   SEC
00064 0061 81 60              CMP A   #$60
00065 0063 26 21              BNE     SETDSP
00066 0065 7F 0005            CLR     SEC
00067 0068 7C 0008            INC     MINAC
00068 006B 96 04              LDA A   MIN
00069 006D 8D 28              BSR     CKDEC
00070 006F 97 04              STA A   MIN
00071 0071 81 60              CMP A   #$60
00072 0073 26 11              BNE     SETDSP
00073 0075 7F 0004            CLR     MIN
00074 0078 96 03              LDA A   HR
00075 007A 8D 1B              BSR     CKDEC
00076 007C 97 03              STA A   HR
00077 007E 81 24              CMP A   #$24
00078 0080 26 04              BNE     SETDSP
00079 0082 86 00              LDA A   #$00
```

```
00119 00D7 11                 CBA
00120 00D8 27 0A              BEQ     X7
00121 00DA 96 0C     X9       LDA A   DEL2
00122 00DC 4C                 INC A
00123 00DD 81 02              CMP A   #$02
00124 00DF 27 15              BEQ     X6
00125 00E1 97 0C              STA A   DEL2
00126 00E3 39                 RTS
00127 00E4 B6 8006   X7       LDA A   PIADB
00128 00E7 84 7C              AND A   #$7C
00129 00E9 D6 10              LDA B   AS2
00130 00EB C4 7C              AND B   #$7C
00131 00ED 11                 CBA
00132 00EE 27 02              BEQ     X8
00133 00F0 20 E8              BRA     X9
00134 00F2 7F 000C   X8       CLR     DEL2
00135 00F5 39                 RTS
00136 00F6 B6 8006   X6       LDA A   PIADB
00137 00F9 8A 02              ORA A   #$02
00138 00FB B7 8006            STA A   PIADB
00139 00FE 96 0D              LDA A   DEL3
00140 0100 4C                 INC A
00141 0101 81 02              CMP A   #$02
00142 0103 27 07              BEQ     X10
00143 0105 97 0D              STA A   DEL3
00144 0107 86 FF              LDA A   #$FF
00145 0109 97 0E              STA A   FLG1
00146 010B 39                 RTS
00147 010C B6 8006   X10      LDA A   PIADB
00148 010F 84 FD              AND A   #$FD
00149 0111 B7 8006            STA A   PIADB
00150 0114 7F 000D            CLR     DEL3
00151 0117 7F 000E            CLR     FLG1
00152 011A B6 8004            LDA A   PIADA
00153 011D 97 0F              STA A   AS1
00154 011F B6 8006            LDA A   PIADB
00155 0122 84 7C              AND A   #$7C
00156 0124 97 10              STA A   AS2
00157 0126 39                 RTS
00158           0000          END
```

Speech Synthesis with Linear Predictive Coding

By Larry Brantingham
Texas Instruments, Inc.



INTRODUCTION

The earliest speech synthesizers were essentially mechanical models of the human vocal tract. Today's computers use two basically different methods to digitize speech. Waveform coding methods produce an output signal that looks like the original voice signal while voice encoder (vocoder) methods use parameters developed from analysis of the speech waveform to simulate the human vocal tract and produce an output that sounds like the original.

One of the vocoder methods, linear predictive coding, is used in an electronic learning aid for spelling produced by Texas Instruments. The Speak & Spell contains a three-chip system consisting of a ROM, a controller, and a speech synthesizer chip. A multistage digital filter, which is part of the synthesizer chip, models the human vocal tract to produce natural sounding speech. In addition to teaching aids, the single-chip speech synthesizer, which is designed to interface with most microprocessors, can be used in communications equipment and consumer products for home and business.

DIGITAL SPEECH CODING

In order to produce electronic speech, digital data representing the spoken word must be first encoded and stored in a memory. The memory then contains information representing words or phrases which are used to reconstruct human speech. The method of encoding employed determines the amount of data needed to represent the speech input and thus affects the size of the memory required for the system. Since many speech coding techniques are used in digital voice communications systems, the data rate used in transmitting information is a convenient means of comparing the data requirements for the different encoding methods. Actually, data rate can be used to mean the rate at which data is transmitted for voice communications or the rate at which data is transferred from memory to the speech synthesizing circuitry.

Current voice waveform coding methods include pulse code modulation (PCM), differential pulse code modulation, adaptive predictive coding, delta modulation and continuously variable slope delta modulation (CVSD). These methods attempt to reconstruct the original speech waveform by digitally representing the amplitude of the voice signal while the signal varies with time. The primary objection to all of these methods is the high data rate necessary for high quality speech coding. Pulse code modulation has the highest data rate with approximately 64 to 100 kbits per second and continuously variable slope delta modulation reaches a low of 16 kbits per second.

Among the vocoder systems currently being used or developed are the channel vocoder, the cepstrum vocoder and the formant vocoder which all utilize models that simulate the human vocal tract with parameters developed from an analysis of the speech waveform. In normal speech, air passing the vocal cords is filtered by the human vocal tract, which includes the mouth, throat and nasal cavity.

Voiced sounds, such as the sound of E in "she," are produced as the vocal cords vibrate. The rate of vibration determines the pitch of the sound generated. Unvoiced sounds, such as the F in "fish," occur when the vocal cords are held open and air is forced past them to the vocal tract.

Vocoder systems thus include parameters relating to amplitude, pitch, stress, and whether the sound is to be voiced or unvoiced. The channel vocoder, for example, uses bandpass filters to divide the speech input into frequency channels. Amplitude, pitch and voiced/unvoiced parameters then control a group of output filters. The cepstrum vocoder uses a Fourier transform to separate the vocal tract resonances from the vocal cord vibrations, while the formant vocoder involves parameters relating to the position of the peaks of the speech frequency spectrum. The advantage of these three frequency based systems is that they have a lower data rate requirement than the waveform coding systems.

LINEAR PREDICTIVE CODING

Computer simulations of the various speech digitization methods have generally shown that the linear predictive methods of digitizing speech can produce speech having greater voice naturalness than the previous vocoder systems. Linear predictive coding, LPC, is a time-based vocoder method that can produce high quality speech with a data rate that is less than 2400 bits per second. LPC uses a recursive filter as a model of the human vocal tract. The basic elements of an LPC voice synthesizer system are shown in the block diagram of Figure 1. The multistage digital filter receives an excitation signal which consists of random pulse sequences representing unvoiced sound or periodic pulses representing voiced sound. As the number of stages in this filter is increased, the speech generated by the system becomes more and more natural sounding. For the TMC 0280 speech synthesizer chip, the digital filter has 10 stages. Tenth order LPC or LPC-10

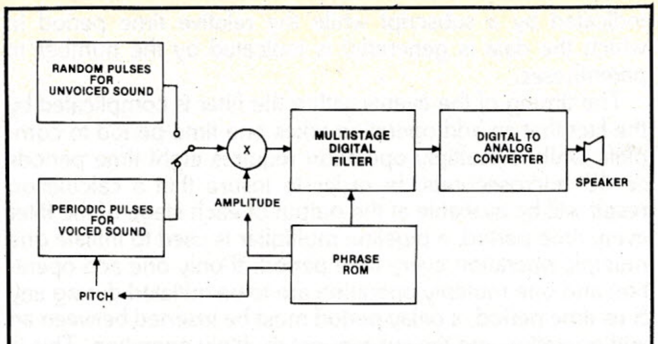


Figure 1. Basic Elements of LPC Voice Synthesis System.

can synthesize speech which is virtually indistinguishable from actual human speech. With a low data rate of 2400 bits per second, the LPC-10 speech analysis/synthesis technique is currently being used for voice communications.

The TMC0350 phrase ROM contains addresses of coded digital data stored in the parameter ROM in the TMC0280 speech synthesis chip. These parameters, which relate to pitch, amplitude, voiced or unvoiced sounds, and filter coefficients, k_1 through k_{10}, are necessary to model the human vocal tract. Pitch and voiced/unvoiced parameters determine whether random or periodic data is supplied to the digital filter. The amplitude or volume of the output signal is determined by an amplification factor that adjusts the constant amplitude signal from the excitation sources.

In modeling the human vocal tract, the excitation signal mimics the function of the vocal cords while the digital filter represents the vocal tract which filters the sounds produced by the vocal cords. The filter coefficients, k_1 through k_{10}, may be thought of as representing the shape or resonances of the vocal tract during speech. Since the vocal tract is constantly changing to produce different sounds, the filter coefficients are also changed or updated periodically. The digital filter outputs a signal which is converted to a synthesized speech waveform by a digital-to-analog converter and this is then connected to a speaker.

DATA STORAGE

The speech synthesizer's design is based on a 50 Hz frame rate, which is the rate at which new speech data is obtained from the phrase ROM. Twelve synthesis parameters, which include the ten filter coefficients, pitch, and amplitude, are stored in the ROM. In order to produce the highest quality speech with the lowest data storage requirements, each of the twelve parameters is allowed to have a certain maximum number of possible values as shown in Table 1.

TABLE 1.

Parameter Number	Parameter	Number of Allowed Values	Number of Code Bits
0	Amplitude	15	4
1	Pitch	32	5
3	k_1	32	5
4	k_2	32	5
5	k_3	16	4
6	k_4	16	4
7	k_5	16	4
8	k_6	16	4
9	k_7	16	4
10	k_8	8	3
11	k_9	8	3
12	k_{10}	8	3
			total 48 bits

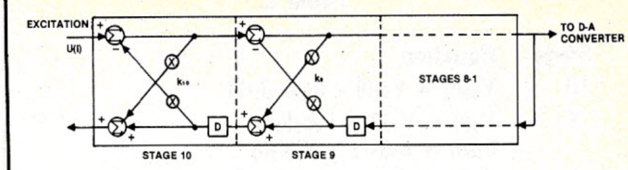


Figure 2. Ten Stage Digital Filter

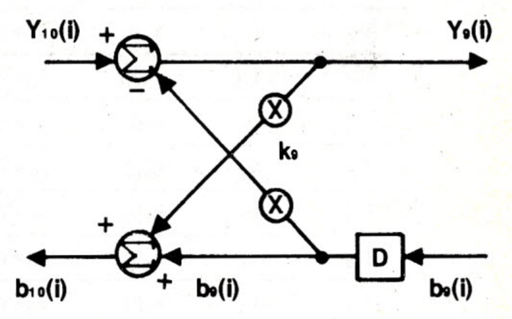
A full set of parameters for each frame would require a data rate of 50 Hz times 48 bits or 2400 bits per second. Three special cases, in which a full frame is not necessary, allow the data rate to be considerably reduced. First, the vocal tract changes shape relatively slowly, so it is often possible to repeat previous coefficients. To accomplish this, a control bit (parameter number 2) is added to each frame. If the control bit is a logic one, no more data is accessed from the phrase ROM and the previous k_1 through k_{10} coefficients are retained.

Second, unvoiced speech requires fewer coefficients. When the pitch parameter is zero, corresponding to unvoiced sound, only k_1 through k_4 are stored in the ROM. Third, natural speech has some pauses between words or syllables. To indicate these pauses, the amplitude parameter is made equal to zero and no other data is required. The combined effect of these three special cases reduces the average data rate to only 1500 bits per second. This low data requirement allows the 131 kbit ROM to store enough data to synthesize approximately 165 words.

THE MULTISTAGE DIGITAL FILTER

The ten stage digital filter, shown in simplified form in Figure 2, has the excitation signal, U(i), applied at the input of stage ten and produces digital data representing synthesized speech at the output of stage one. Each stage of the filter contains two multipliers, two adders, and a delay circuit, which consists of a shift register. The timing of events within the filter relates to the 10 kHz sample rate used in sampling the original speech. Within the 100 μs sample period, each stage of the filter must perform two multiplications and two additions for each digital sample to be generated.

The calculations performed in each stage are represented by the illustration of stage 9 in Figure 3. The first equation shows that the filter coefficient, k_9, is multiplied by the output of the delay circuit and the resulting product is then subtracted from the input, $Y_{10}(i)$, to yield the output, $Y_9(i)$. This output is used as part of the feedback to stage ten. To obtain the feedback, $Y_9(i)$ is multiplied by k_9 and the product is subtracted from the output of the delay circuit, $b_9(i-1)$, to yield $b_{10}(i)$. Equations for the various stages are summarized in Table 2.



$$Y_9(i) = Y_{10}(i) - k_9 b_9(i-1)$$
$$b_{10}(i) = b_9(i-1) + k_9 Y_9(i)$$

Figure 3. Calculations within a Single Stage

Table 2.

Stage	Equation
10	$Y_{10}(i) = Y_{11}(i) - k_{10}b_{10}(i\text{-}1)$
9	$Y_9(i) = Y_{10}(i) - k_9b_9(i\text{-}1)$
	$b_{10}(i) = b_9(i\text{-}1) + k_9Y_9(i)$
8	$Y_8(i) = Y_9(i) - k_8b_8(i\text{-}1)$
	$b_9(i) = b_8(i\text{-}1) + k_8Y_8(i)$
•	•
•	•
•	•
1	$Y_1(i) = Y_2(i) - k_1b_1(i\text{-}1)$
	$b_2(i) = b_1(i\text{-}1) + k_1Y_1$
	$b_1(i) = Y_1(i)$, which is the output of the filter.

In the operation of the filter, the 100 μs sample period is divided into 20 5μs time periods, T_1 through T_{20}. The operations in each of the 10 stages cannot be accomplished simultaneously since Y_{10} must be calculated before Y_9, which must be calculated before Y_8, etc. during any given time period. Also, the b_{10} through b_1 data must be calculated during the same time period and stored in the delay circuit of each stage for use during the next time period. In Figure 3 and Table 2, the stage in which the Y and b data are used is indicated by a subscript while the relative time period in which the data is generated is indicated by the number in parentheses.

The timing of the events within the filter is complicated by the fact that an add operation takes one time period to complete while a multiply operation requires eight time periods or 40 microseconds. In order to insure that a calculation result will be available at the output of each stage of the filter every time period, a pipeline multiplier is used to initiate one multiply operation every time period. If only one add operation and one multiply operation are to be initiated during any 5 μs time period, a delay period must be inserted between an add operation and the subsequent multiply operation. This is the reason for the delay circuit shown in Figure 2. With the proper timing, operations in all the stages of the digital filter are performed concurrently and appropriate intermediate results become available as needed.

The lower output from stage 10 of the digital filter is the intermediate result, b_{11}, which is not used in the operation of the filter. Thus, in this stage, one multiplier and one adder could be deleted. The multiplier, however, is used to control the amplitude of the output signal. The excitation signal for the production of voiced or unvoiced is scaled by an amplification factor, A. During the time that the product, $k_{10} \bullet Y_{10}(i)$, (which is part of b_{11}) would otherwise be calculated, the amplification factor is multiplied by the excitation data.

Table 3.

Time Period	Multiplier Inputs from k stack	Multiplier Inputs from Data Bus	Multiplier Output & One Adder Input	Other Adder Input	Adder Output	One Period Delay Output	Shift Register Output	Y Latch Output
T_1	$-k_2$	$b_2(i\text{-}1)$	$-k_{10}\bullet b_{10}(i\text{-}1)$	$U(i)$	$b_2(i\text{-}1)$	$b_3(i\text{-}1)$	$b_{11}(i\text{-}1)$	$Y_1(i\text{-}1)$
T_2	$-k_1$	$b_1(i\text{-}1)$	$-k_9\bullet b_9(i\text{-}1)$	$Y_{10}(i)$	$Y_{10}(i)$	$b_2(i\text{-}1)$	$b_{10}(i\text{-}1)$	•
T_3	A	$E(i+1)$	$-k_8\bullet b_8(i\text{-}1)$	$Y_9(i)$	$Y_9(i)$	$Y_{10}(i)$	•	•
T_4	k_9	$Y_9(i)$	$-k_7\bullet b_7(i\text{-}1)$	$Y_8(i)$	$Y_8(i)$	$Y_9(i)$	•	•
T_5	k_8	$Y_8(i)$	$-k_6\bullet b_6(i\text{-}1)$	$Y_7(i)$	$Y_7(i)$	$Y_8(i)$	•	•
T_6	k_7	$Y_7(i)$	$-k_5\bullet b_5(i\text{-}1)$	$Y_6(i)$	$Y_6(i)$	$Y_7(i)$	•	•
T_7	k_6	$Y_6(i)$	$-k_4\bullet b_4(i\text{-}1)$	$Y_5(i)$	$Y_5(i)$	$Y_6(i)$	•	•
T_8	k_5	$Y_5(i)$	$-k_3\bullet b_3(i\text{-}1)$	$Y_4(i)$	$Y_4(i)$	$Y_5(i)$	•	•
T_9	k_4	$Y_4(i)$	$-k_2\bullet b_2(i\text{-}1)$	$Y_3(i)$	$Y_3(i)$	$Y_4(i)$	•	•
T_{10}	k_3	$Y_3(i)$	$-k_1\bullet b_1(i\text{-}1)$	$Y_2(i)$	$Y_2(i)$	$Y_3(i)$	•	•
T_{11}	k_2	$Y_2(i)$	$U(i+1)$	0	$Y_1(i)$	$Y_2(i)$	$b_{10}(i\text{-}1)$	•
T_{12}	k_1	$Y_1(i)$	$k_9\bullet Y_9(i)$	$b_9(i\text{-}1)$	$U(i+1)$	$Y_1(i)$	$b_9(i\text{-}1)$	•
T_{13}	$-k_{10}$	$b_{10}(i)$	$k_8\bullet Y_8(i)$	$b_8(i\text{-}1)$	$b_{10}(i)$	$U(i+1)$	$b_8(i\text{-}1)$	•
T_{14}	$-k_9$	$b_9(i)$	$k_7\bullet Y_7(i)$	$b_7(i\text{-}1)$	$b_9(i)$	$b_{10}(i)$	$b_7(i\text{-}1)$	•
T_{15}	$-k_8$	$b_8(i)$	$k_6\bullet Y_6(i)$	$b_6(i\text{-}1)$	$b_8(i)$	$b_9(i)$	$b_6(i\text{-}1)$	•
T_{16}	$-k_7$	$b_7(i)$	$k_5\bullet Y_5(i)$	$b_5(i\text{-}1)$	$b_7(i)$	$b_8(i)$	$b_5(i\text{-}1)$	•
T_{17}	$-k_6$	$b_6(i)$	$k_4\bullet Y_4(i)$	$b_4(i\text{-}1)$	$b_6(i)$	$b_7(i)$	$b_4(i\text{-}1)$	•
T_{18}	$-k_5$	$b_5(i)$	$k_3\bullet Y_3(i)$	$b_3(i\text{-}1)$	$b_5(i)$	$b_6(i)$	$b_3(i\text{-}1)$	•
T_{19}	$-k_4$	$b_4(i)$	$k_2\bullet Y_2(i)$	$b_2(i\text{-}1)$	$b_4(i)$	$b_5(i)$	$b_2(i\text{-}1)$	•
T_{20}	$-k_3$	$b_3(i)$	$k_1\bullet Y_1(i)$	$b_1(i\text{-}1)$	$b_3(i)$	$b_4(i)$	$b_1(i\text{-}1)$	$Y_1(i\text{-}1)$
T_1	$-k_2$	$b_2(i)$	$-k_{10}\bullet b_{10}(i)$	$U(i+1)$	$b_2(i)$	$b_3(i)$	$U(i+1)$	$Y_1(i)$

DIGITAL FILTER OPERATION

The equivalent of a 10 stage digital filter, when implemented in MOS large scale integrated circuitry, has the form illustrated by the block diagram in Figure 4. The filter coefficients, k_1 through k_{10}, and the amplification factor are stored in the k-stack. The k-stack consists of 10 shift registers each of which has 10 stages so that each of the 10 coefficients can be circulated for 10 time periods. This insures that the correct coefficients for each multiply operation are always available. The pipeline multiplier performs the multiplications for each stage of the filter. The multiplier receives k_1 through k_{10} and A from the k-stack and it receives Y and b data from other parts of the filter via the data bus. The multiplier initiates a new multiply operation once every five microseconds.

Since each of these operations requires 40 microseconds to complete, the multiplier has eight stages. Digital multiplication is accomplished by a series of addition and shift operations as the data propagates through the eight stages. Since a new multiplication operation is initiated every five microseconds, there are eight multiplications in progress at any given time, and the multiplier completes one operation each 5 μs time period.

The various intermediate results occurring in the different parts of the digital filter are listed in Table 3. The transfer of data from one part of the filter to another is controlled by logic switches which are illustrated by mechanical switches in Figure 4.

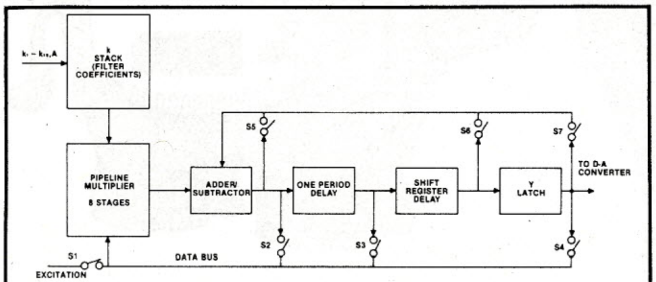


Figure 4. Block Diagram of Digital Filter

The multiplier receives two inputs; one is from the k stack while the other varies according to which of the switchés S2, S3, or S4 is closed. At time period T1, the mutliplier receives an input from the adder/subtractor circuit via switch S2. At time period T2, S4 is closed and the multiplier receives an input from the Y latch. At time period T3, the multiplier receives the excitation signal, E(i+1), via switch S1. During time periods T4 through T12, the multiplier's input is the output of the one period delay circuit and from T13 through T1 of the next cycle, the multiplier's input is the output of the adder/subtractor circuit.

The adder/subtractor circuit receives two inputs: one from the multiplier and, depending on the time period, one from switch S5, S6 or S7. During time periods T2 through T10, the adder receives its own output as an input via switch S5. From T12 through T19, switch S6 connects the output of the shift register to the input of the adder; and at time period T20, switch S7 connects the output of the Y latch to the adder. The output of the adder is 14 parallel bits which is delayed one time period and then applied to the shift register. The shift register circuitry actually consists of 14 shift registers to accommodate the 14-bit wide data from the one period delay circuit. Each shift register consists of eight stages which provide eight time periods of delay required for the multiply operation. The synthesized speech output of the filter is the output of the shift register at time period T20. The Y latch stores this output for one cycle so that it is available as an input to the adder via switch S7.

THE SPEECH SYNTHESIZER CHIP

In addition to the digital filter, the TMC0280 speech synthesizer chip contains a parameter storage RAM, a decoding ROM, excitation logic, interpolation logic, timing logic, interface circuitry, and a digital-to-analog converter.

The excitation to the digital filter may take two forms. Sounds which have a definite pitch, such as vowel sound or voiced fricatives (Z, B, D, etc.) require a periodic input function. Unvoiced sounds (S, F, T, Sh, etc.) require a white noise source. For voiced sounds, a 5 millisecond chirp is applied to the input of the filter at a time interval equal to the pitch period. The chirp is stored in a 50 x 8 chirp ROM which is addressed by the pitch period counter. During a voicing transition and prior to the start of speech, special provisions are made to zero the pitch counter. For unvoiced sounds, the excitation has a constant magnitude of 0.5 and a random sign, which is provided by a 13-bit shift counter with three Exclusive OR feedback circuits.

In the operation of the system, speech parameters are transferred from the TMC0350 phrase ROM to the speech synthesizer where they are stored. One complete set of 12 parameters, which are used as target values during parameter interpolation, is stored in the parameter RAM. This RAM has a variable word length in order to accommodate the speech parameters; as shown in Table 1, the pitch parameter requires five code bits, while the filter coefficients may have three, four or five bits. Data is supplied to the RAM via the parallel outputs of a serial shift register which accepts data from the phrase ROM. The parameter RAM outputs are used as inputs to the parameter ROM.

The frame of data stored in the parameter RAM consists of 12 codes or pointers which select 10-bit parameter values from a 216 x 10 bit ROM. Each speech parameter in this synthesizer system requires its own lookup table, so the parameter ROM is organized as 12 ROMs of 2^n bits where n is the number of code bits for a given parameter. The 10-bit parallel output of the ROM is applied to the inputs of a parallel-to-serial converter, which is loaded according to timing signals generated by the interpolation logic.

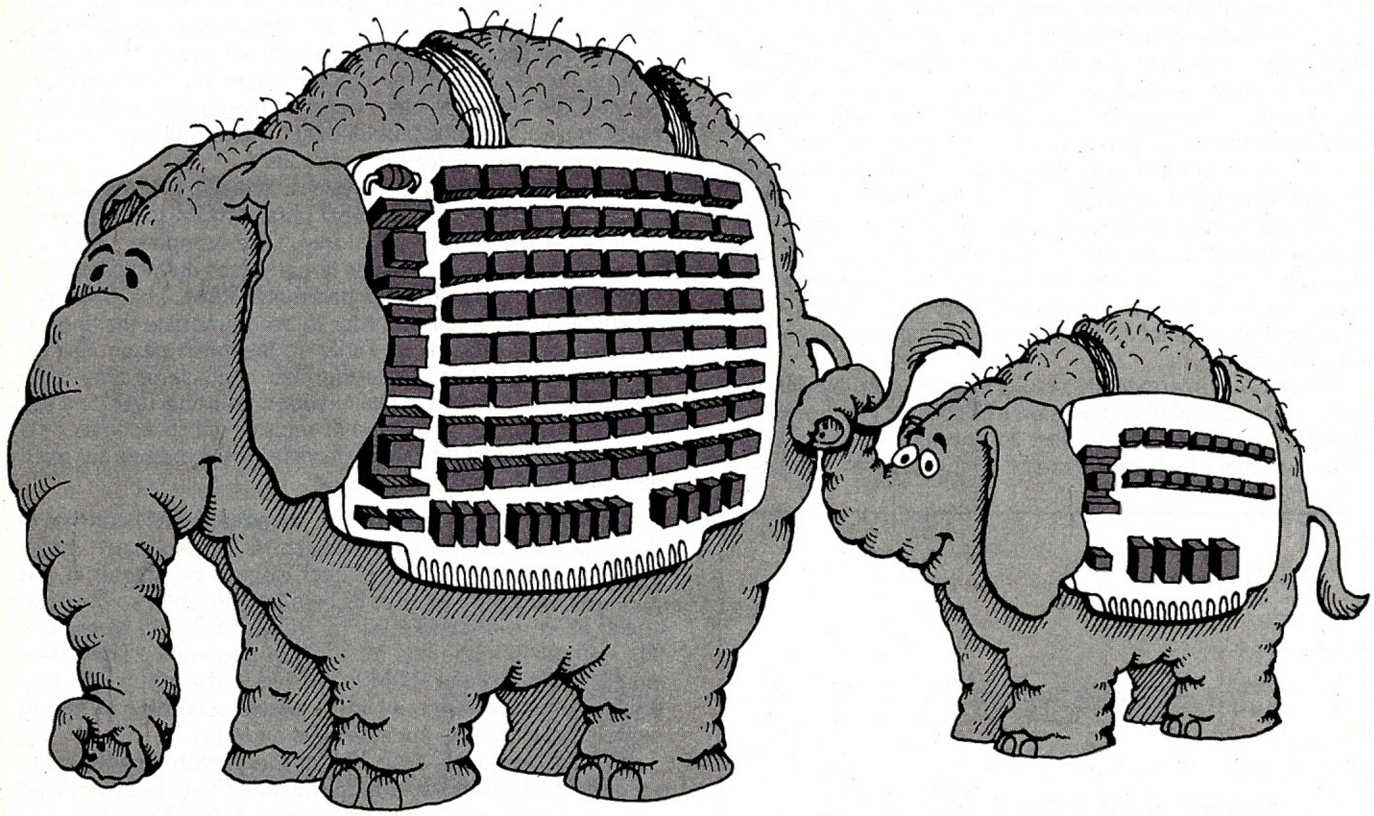
In most cases, it is desirable for the speech parameters to vary smoothly from frame to frame rather than to be updated only at the 20 ms frame period. The TMC0280 chip contains logic circuitry to do an approximately linear interpolation of all 12 parameters at eight points within the frame or once each 2.5 ms. The parameters are interpolated one at a time as selected by the parameter counter. To conserve area and eliminate errors, the interpolation logic calculates a new parameter from the present value and the next or target value stored in code in the parameter RAM.

As the speech parameters are decoded and interpolated, they are used in the operation of the digital filter. The output of this filter must be converted to analog form before being connected to a speaker. The chip contains an 8-bit digital-to-analog converter with an accuracy of one half the least significant bit. The circuit also contains low-pass filtering and 100-mW push-pull speaker drivers, which eliminate the need for an external amplifier. During the operation of the circuit, the output of the filter (Y1) is updated every 100 μs. The direction of current flow in the 100-ohm speaker is determined by the sign of Y1, which accomplishes the D-to-A conversion for the most significant bit. Filtering is provided to eliminate effects of the 10-kHz sample rate and to de-emphasize the output signal in compensation for the pre-emphasis applied in the analysis of the original speech.

CONCLUSION

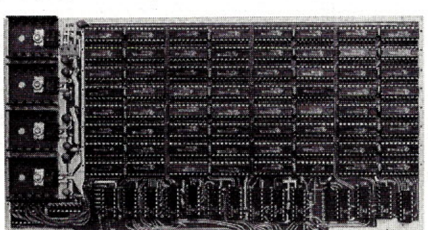
The design goals for the TMC0280 speech synthesizer chip included the capability to produce high quality speech at a low data rate, low system cost with minimum external components, use of a low-cost fabrication process such as metal gate PMOS, flexibility to process data internally,and a simple external interface. All of these goals have been achieved and while talking computers are not yet a household item, this speech synthesizer chip brings us one step closer.☐

S-100 Memory for peanuts, Artec has slashed its prices!



32K for $620, 16K for $315–Assembled.

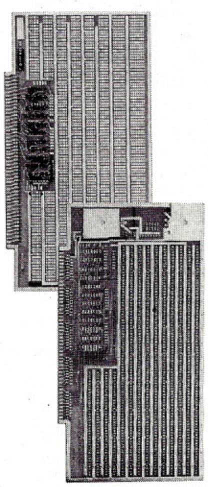
No matter what your needs, Artec has a memory board for you. You can start with 8K of TI 4044 memory on a 5.3″ x 10″ card and work your way up to a full 32K in 8K increments. The access time is only 250ns. The memory is addressable in 4K blocks and is perfect for S100 and battery augmented systems. The Artec 32K Expandable Memory has four regulator positions, bank select and plenty of room for all necessary support hardware. It uses less than 1 amp per 8K of memory (3.9 for 32K), and only +8 volts.



BOARD	KIT	ASSEMBLED
8K Memory	$150	$175
16K Memory	$265	$315
24K Memory	$400	$475
32K Memory	$520	$620

Add-ons: $135—Chips alone: $7.00

GP100—$20.00
Maximum design versatility along with standard address decoding and buffering for S100 systems. Room for 32 uncommitted 16 pin IC's, 5 bus buffer & decoding chips, 1 DIP address select switch, a 5 volt regulator and more. High quality FR4 epoxy. All holes plated through. Reflowed solder circuitry.



WW100—$20.00
A wire wrap breadboard, similar to the GP100. Allows wire wrap of all sizes of sockets in any sizes of sockets in any combination. An extra regulator position for multiple voltage applications. Contact finger pads arranged for easy pin insertion.

Buffering Kit—$12.65
All the necessary components to bootstrap any Artec board into your system. Buffering I/O, DIP switch heat sinks and every support chip you need.

TO ORDER: Use your Mastercharge or BankAmericard. Or just send along a money order. We can accept only U.S. currency. Please include $3 handling on all orders. California residents add 6.0% sales tax.

FOR MORE INFORMATION: For more information about these or any of Artec's complete line of circuit boards or for either industrial or personal use, please call or write. A catalog will gladly be sent.

Please send me: (Include Quantity)
___ 32K ___ GP100 ___ WW100
☐ I've enclosed a money order.
☐ Mastercharge No. ______ ______ Exp. Date
☐ BankAmericard No. ______ ______ Exp. Date
Name ______
Address ______
City ______ State ____ Zip ____
Calif. Res. add 6% Sales Tax *$3.00 Handling Encl.*

ARTEC ELECTRONICS, INC.
605 Old County Rd., San Carlos, CA 94070
(415) 592-2740

Software Design for Small Business Systems

By Robert C. Mooney

The two major problems encountered in custom software generation are: the user does not know, is not sure, or cannot specify what he wants; and programmers often begin coding of the program immaturely without adequate planning, or get unnecessarily involved in details.

Although software design is referred to as the "fun" job, anyone who has done a large amount of programming will know that it is not always as easy as it seems.

Once the desired hardware and systems software have been selected, we may proceed with the assumption that the user has at least a general idea of what goals and objectives he wishes to implement by means of the computer. These system objectives should be considered in light of the power of the hardware. Costly extravagances should be eliminated and decisions made as to what must be implemented immediately and what expected improvements will come later.

It is important that the system operate in a manner that the end user can understand and is familiar to him. The end user is the one who must operate and live with the system, not the programmer.

DEFINE AVAILABLE INPUTS AND DESIRED OUTPUTS

The inputs which are to be available to the system must be outlined and defined. If appropriate forms are currently in use, samples should be obtained to aid in the definition of inputs. If there are no forms available, the general format of input data should be drafted in interaction with the user.

If there is a manual accounting system in operation, begin with the operation of the manual system. Even if the existing system has deficiencies, it will at least provide a basis for the programmer's understanding of the operation. Decide what improvements need to be made, but do not attempt to make drastic reorganizations all at once.

What printed reports are desired? What are the screen displays desired? The desired outputs must be determined as completely as possible so that the necessary data items may be incorporated into the design of the file structure.

FILE STRUCTURE DESIGN

The general file requirements will become clear as the inputs and outputs are finalized. When dealing with the small to medium sized system, it is necessary to precisely define required file sizes and record sizes within the files. This is because required files (including program files) must be organized on a finite amount of mass memory. Most systems on the market today employ the magnetic disk as the primary mass storage media, and the required file sizes will determine what computer functions may be on-line simultaneously and what functions will require the exchange of the disk media.

If the disk is hard sectored or if utilization of random access files is made, the sector size or logical block size of the related disk drive and DOS should be considered. At least a reasonable effort should be made to cause each record to occupy an integral number of blocks or sectors; or for small records, to ensure that a block will contain an integral number of records with a minimal waste of storage space.

For example, if there are 512 bytes in a block and the record size is 22 bytes, there will be 23 records in each block with six bytes remaining unused at the end of each block.

File structure tables are the most important form of documentation for both the coding of the program and for future software modifications, especially when the coding is to be done by more than one person. A file table shows each of the data items contained in a record of the file and the data type and estimated length in bytes of each data item.

In order to obtain the required output reports, all required data must be stored in the logical file. It is often a good idea to store as much of the input data as possible even though there is no immediate requirement, storage permitting, of course.

AN EXAMPLE

As an example of program design, we may consider the implementation of a simple purchase order system. The outcomes expected from the purchase order system are as follows:

- To maintain strict control and recordkeeping of payments made to vendors.
- To provide for a numerical listing of purchase orders showing the total amount owing.
- To provide for a purchase ledger file showing all expenditures for year-to-date and for the current period by ledger account number.

What are the given inputs? Obviously, there is input of purchase orders. We visit the user's site and obtain copies of his purchase order, or establish preliminary designs for the purchase order form.

Another input which will be required is input to a checks file for payments to vendors. It may be desirable to review the purchase order file, determining which amount will be paid now and which amounts will be deferred. Or it may be desirable to enter the amounts to pay directly by the vendor or purchase order identification.

The desired outputs will be the *Open Purchase Order Register,* the *Disbursements Journal* which shows all payments during the period, and the *Expense Ledger* which shows all purchases for the current period and year-to-date by ledger account.

We then determine that there are four files which will be required:

- Purchase order file
- Checks or disbursements file
- Ledger file
- Paid purchase order or expense history detail file

Adjustments may have to be made with the expense history detail file. If the disk or mass storage facilities of the system are limited, the expense histories may be printed in hard-copy form periodically and the pages filed manually. If an accounts payable system is incorporated, it may be desirable to list all expenditures by each vendor identification. In a system with moderate disk capacity it should be possible to maintain a complete year's history of expenses.

We see that the possible program functions which will be required will be:

- Enter purchase orders
- Verify goods received
- Enter payments (checks)
- Print checks
- Open purchase order report
- Disbursements journal print
- Expense ledger listing

If the system is to be menu structured, we now have the preliminary menu display for the purchase order system.

Then, beginning with the Purchase Order file, we create a file structure table as follows:

Purchase Order File

Description	Maximum Length
P.O. number	7
Invoice number	7
Originated by	3
Date of P.O.	6
Approved by	3
Goods received (Y/N)	1
Vendor name and address	90*
Description of purchase	20
Ledger account number	7
Amount paid	7
Amount outstanding	7
Discount amount	7
Terms	20
Total bytes	185

This purchase order file requires 185 bytes for each P.O. item on file. Since the disk drive and DOS of the target system is structured in 256 byte blocks for random files, (which may or may not be the case depending upon the actual hardware used) this leaves 256 − 185 = 71 bytes wasted at the end of each block.

In order to make more efficient use of this disk space, the decision is made to eliminate the vendor name and address* from the P.O. file, inserting instead the vendor name (30 bytes) and leaving the vendor address to be found from a vendor file or mailing label program to be added to the system. This reduces the length of the purchase order record to 125 bytes per P.O. Since the DOS for our system will insert two records into one 256 byte block, the 6 unused bytes in each block may be tolerated.

OPTIMIZATION

Of course, a reasonable effort should be made to make the software system as efficient as possible. The programmer is cautioned, however, to refrain from spending too much time on software optimization at this stage. Invariably changes will be discovered that must be made before the job is done, and some of the most elaborate schemes may have to be redone anyway.

It is more important to strive for *working* software and keeping things simple. The simpler the design, the easier it will be for the coding of the program and the subsequent debugging of the software. You will find eventually that the system will become complicated all by itself because even the simplest form of real life is intricate.

Avoid wasting time on unnecessary extravagances. The cost of coding some complicated enhancements may outweigh the benefits to be achieved. Also remember that the software system must be easily understood by the personnel who operate it. It is better to have a few simple programs which are frequently used than an expensive myriad of seldom used software.

Note that the example P.O. file contains a flag to be inserted as Y(yes) to verify that the ordered goods have been received. It may be desired to enter the vendor's invoice number in the field provided when the items are received, leaving the invoice number equal to zero or null to indicate that the goods have not yet been received. This will further reduce the P.O. record length by one byte.

The P.O. file also contains the facility for entering an applicable discount amount for the P.O. and two fields for amount paid and amount outstanding. The paid and outstanding items will provide for the possibility of making partial payments. The addition of amount paid and amount outstanding and discount amount (−) gives the total amount of the P.O. A similar design is used for the ledger file:

Ledger File

Description	Maximum Length
Ledger account number	7
Description	30
Month to date amount	8
Year to date amount	8
Annual budget amount	8
Total bytes	61

This provides for four ledger account entries per block with 12 bytes unused. The amount items are each 8 bytes in length. This provides for a dollar amount of $999,999.99 (if stored without the punctuation) or up to one million dollars, which should be sufficient for most applications.

The checks (disbursements) file, and the history detail file then follow:

Detail File

Description	Maximum Length
Ledger account	7
P.O. number	7
Check number	7
Date	6
Vendor name	30
Amount	7
Total bytes*	64

And finally, the file requirements of the purchase order system is determined as follows:

Purchase order file:			
128 bytes times	50 P.O.'s	=	6400
Checks file:			
64 bytes times	50 checks	=	3200
Ledger file:			
64 bytes times	50 accounts	=	3200
History detail file:			
64 bytes times	600 items	=	38400*
File storage bytes total:			51200

This provides for up to 50 P.O.'s and checks per month with a complete year's disbursements detail history stored on the disk and printed out and cleared at year's end. There is also ample space remaining on the disk for program storage, and spares for last minute changes. At a later date a vendor file and another history detail file may be included by vendor number as well as by ledger account number on the same disk.

Now the program functions may be outlined in more detail. The programs which will be required are:

- Enter Purchase Orders
- P.O. inquiry to verify goods received, O.K. to pay, etc.
- Review open P.O.'s entering amount to pay, generating checks file
- Print checks
- Print disbursements journal (check register) removing paid P.O.'s from open P.O. file, entering checks to detail history file
- Print open purchase order report
- Ledger account inquiry
- Expense ledger listing

The first two items may possibly be combined and the fifth item may be broken down into two separate phases to pro-

*By the use of ASCII or radix 50 packing techniques, the detail history file item may be reduced to a length of 32 bytes each, thus doubling the number of transactions which may be stored. However, this will significantly increase program storage requirements and program development time and is not recommended unless disk storage is at a severe premium.

vide for verification of the checks file before the updates are made to the open P.O. file and the detail history file. A program to print the detail history file (probably in ledger account number order) and to clear the transactions from the history file and zero out the year-to-date totals in preparation for the next year will be required.

The next project is to determine the finalization of report format specifications to ensure that all required data items will be entered and stored in the proper files.

DOCUMENTATION

The generation of good documentation is essential to the programming process. The principles of operation of the software should be considered very carefully and everything should be written down. Even if you are to be the sole programmer on the project, you will often forget how the program works if program changes are required months or even years later.

Do not attempt to begin the actual coding of the program until the design has been considered and reconsidered very carefully. The preliminary documentation generated at this stage will greatly ease the actual coding of the program and future program changes, especially if multiple programmers will be working on the job.

CODING THE PROGRAM

If the system design has been done properly, the actual coding of the program is the easiest part of the job. It is important to use accepted good programming techniques. Simple straight-line programs are easier to debug and maintain than "tricky" programming. Commonly used logical procedures should be organized into subroutines or even separate program overlays.

It is not possible to take the file structure tables devised during the software design phase and fill in the actual variable names to be used in the program code. If at all possible, the same variable names should be used for the same data items throughout the entire software to avoid confusion.

In some systems the length in bytes of each data item stored on disk is very critical. The programmer must ensure that this length is controlled to guarantee compatiblity of data files generated by each program.

Try to stick to your original plan when coding the program. You will only confuse yourself by trying to change the designs of the software while doing the coding. If you must make changes in the initial program design, each change must be considered carefully to determine the full extent and effects of the change. Remember to keep your documentation up to date. The best way to do this is to change the documentation **first.**

Use remark (comment) statements in the programs as much as available memory permits, and again, do not spend excessive time making the programs optimal; you will get involved in details and lose sight of the general plan of the program. First the program must be operative, then gradually the inefficiencies may be weeded out.

In the simple purchase order system, the addition of an accounts payable system with vendor files at some future date has been anticipated. Therefore, it is necessary to bear in mind that the vendor name and address in the P.O. file may be replaced by the vendor number only, with the vendor name being determined by reference to the vendor file. To facilitate this future link to the accounts payable system, remark (comment) statements should be inserted in the purchase order programs to indicate the location of anticipated reference to the accounts payable files.

DEBUGGING AND SYSTEM INTEGRATION

If the programming was structured into logical modules, the checkout phase will be much easier. Program segments and subroutines may be tested independently as the code is written. During the testing phase, the program should provide for the printout or display of critical data as the program operates. Breakpoints or halts may be temporarily inserted into the program so that its operation may be verified at critical points. Also, cross-checks may be coded into the program so that some errors are automatically detected.

A testing curriculum should be established so as to check the entire software as completely as possible. For large programs it may be desirable to document a detailed acceptance testing procedure. The final phase of program checkout includes the complete verification of the software with live user data. This is essential for proper programming because the problems which may arise in real life operation can never be approximated by hypothetical test cases.

As the debugging phase nears completion and the program appears operative, the system integration phase may begin. Operations manuals and documentation on all phases of the system must be drafted and operator training initiated. Some installations may require several levels of documentation: beginning user, experienced user, and detailed software descriptions for future software changes. Good documentation is essential so that someone unfamiliar with the system will be able to operate it if necessary. Also, you as the programmer may avoid being constantly disturbed by unnecessary questions if adequate documentation is provided.

ADDITIONAL SUGGESTIONS FOR BETTER PROGRAMMING

All data inputs should be checked by the computer for proper data type (alpha or numeric) and proper length of data and immediate operator feedback generated in case of error.

The operator should be given the opportunity to check input transactions data and acknowledge that all items are correct before updates are made in the files.

Audit listings of all transactions should be generated and kept on file for backup reference and as a constant record of system operation.

If the same variable names are used for the same data items in all programs, common subroutines may be used in different programs with no confusion.

A file maintenance program should be provided for each file in the system. This program should include the capability of listing the entire file on the printer, display of the file on the display screen, or to display an individual item of the file to make corrections in the record, delete records from the file, or add new records to the file.

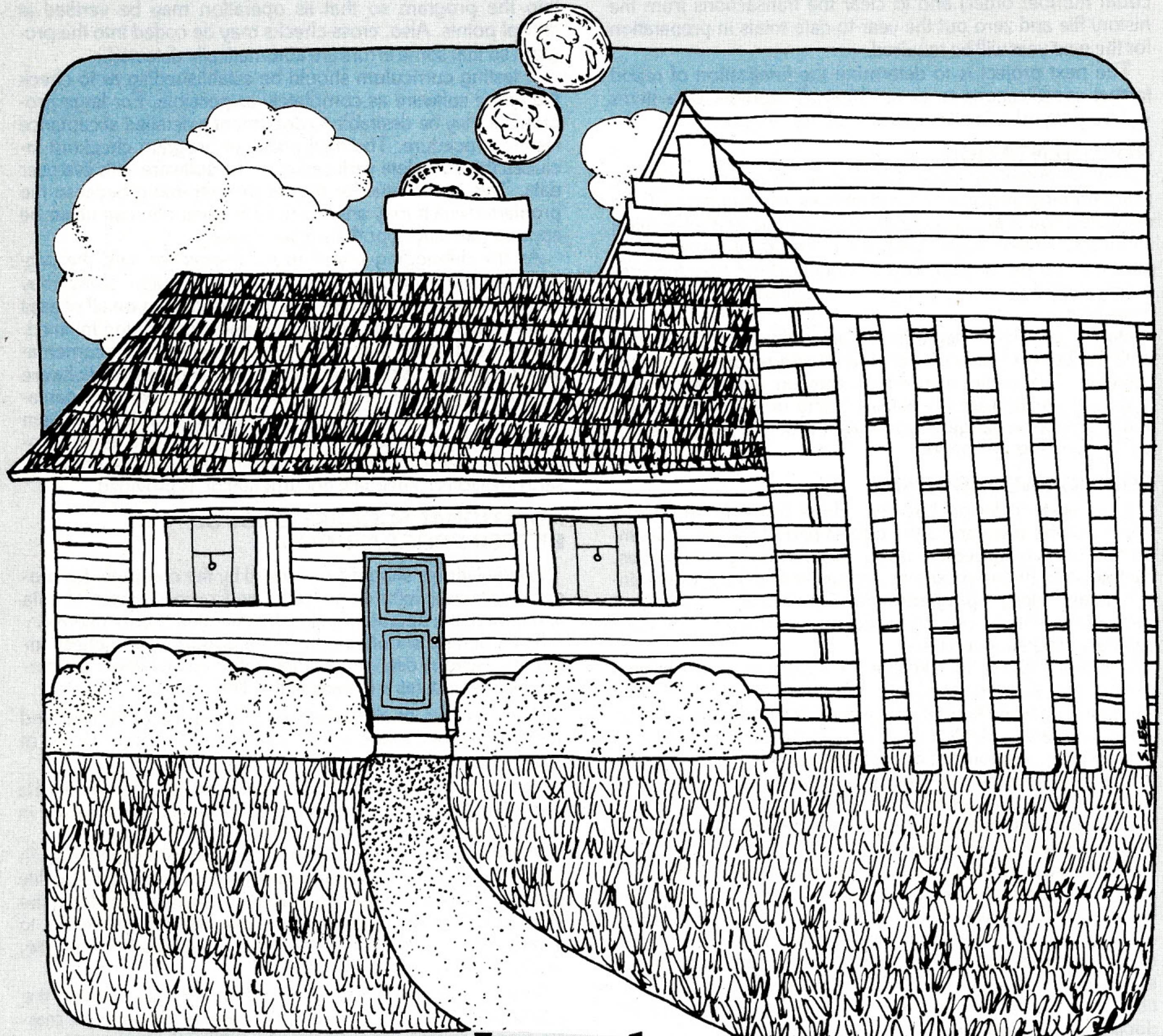
The operator should be provided the means of halting the operation of any program at any time and returning to the master menu selector without having to re-boot the entire system.

FINAL NOTES

It is generally accepted that all large programs contain minor bugs which may show up periodically. Often the inconvenience caused is so trivial that it is not worth the time required to locate and correct it. The process of software development may continue over a long period of many tests and adjustments, and software enhancements sometimes continue to be developed over a period of years. Users and programmers should consider all proposed software changes carefully and avoid making spurious changes until all possible aspects have been considered. Often it may be necessary to retest and recertify the entire software after making a seemingly small change.

BACKUP PROCEDURES

Backup copies of both data and program files should be made on a regular basis. A man-year investment in software should be well worth the small cost of backup copies. It is often desirable to maintain a third backup copy of the system at a secure site separate from the primary computer installation.□



Lumber

By Jim Schreier, Associate Editor

Adding an addition or remodeling is much less expensive than moving into a larger home. Lumber and building materials are expensive, but sharp shoppers can reduce the sting of inflation. This program can help. It was designed to show that those with microcomputers can save expense and energy with accuracy either as a businessman or consumer.

LUMBER calculates board footage, square footage, cost per both 100 and each of 35 items; figures respective costs, adds sales tax and dray and gives a net total. The program is presented in the format of a lumber and materialman's bid, similar to bids lumberyards give on house or remodeling packages. Since the format is general in nature, even calculating the board footage and cost of home woodworking projects is as fast as entering the information at a terminal. Other lumberyards may wish to adapt this format to their needs, while contractors will be able to double check bids for accuracy. Standard arithmetic procedures utilized by most lumberyards are used within the program.

PROGRAM LAYOUT

LUMBER is about 5400 bytes and is written in SWTPC's standard 8K BASIC. Effort was made to use only a minimum of the SWTPC BASIC features to allow hassle-free conversion to other string-handling BASICs. About 2500 bytes

are reserved in the DIM statements with string length set to 28 (Line 70). If your BASIC does not have the capacity of setting string lengths, additional RAM may be necessary. Formatting is based on the SWTPC CT-64 terminal system using 16 line x 64 columns. Two clear screen and home up commands are used at lines 40 and 250. These lines currently call for a control Z and control B. Alter these lines to meet any specific needs.

LUMBER has three divisions: Introduction, printout and calculations.

Line 70 formats maximum column length (LINE=120) as well as 2 digits to the right of the decimal point (DIGITS=2). Both statements are optional. DIGITS=2 may be formatted with PRINT USING statements found in other BASICs. Line 60 calls for FUNCTION A to format digits in justified columns.

To use LUMBER without the bid headings, remove lines 80-130, 260-390 and 660-780.

The printout occurs from lines 400-790. You may wish to direct this section to a printer. It is currently programmed only to be displayed on the CRT. This section may be adapted to store the bid information on floppy disk as well.

The five calculations are found at lines 800-1830.

Which is cheaper, 40 2x4 #2 studs at $1.89 each or at $250 per M board feet?* The cost per each is simple multiplication. Actually, so is board footage. The purpose of board footage is to reduce any given lumber to a constant one foot of material one inch thick. Although a century ago lumber was a full inch thick, today it is reduced to about ¾". It is nevertheless calculated as if it were an inch thick. The formula for conversion is:

$$BF = \frac{LxW}{12} \times LF$$

with L being the length, W being the width and LF being the lineal or running footage. The LxW/12 produces a constant conversion factor. A 2x4 has the factor of .666666, a 1x12 is 1.0, etc.

Lineal footage is calculated by multiplying the number of pieces by the length. It should be noted that length is usually figured by rounding up from the nearest six inches. A 93⅝" 2x4 is figured as a full eight feet.

LUMBER calculates each board footage ratio rather than using DATA/READ statements. This not only saves memory but allows for odd and more unusual board footage calculations. Traps are located in lines 900 and 910 to prevent entering a full sized tree by accident. The program balks at a 100x100 entry for good reason. Of course, the user has final say, however, as the program calls the unusual size to his attention. The operator must decide if the correct input has occurred. The trap is sprung if the ratio is less than .1666667 or greater than 8.0.

Most lumber is sold in even length units. Lumber less than six feet, greater than 24 feet or in odd footage sizes is designed to spring another trap (Lines 860-880). The coding to determine odd or even footage is:

```
IF X/2<>INT(X/2) THEN . . .
```

Again, there is the option of overriding the length trap.

Board footage (BF) and board footage through lineal footage (LF) will handle most lumber calculations, however, additional calculations are necessary for other building materials.

Paneling, sheetrock (also known as plasterboard, etc. depending on what part of the country you call home), insulation, plywood, hardboard, building board and particle board are calculated by square footage (SF). The two most common sizes are 4x8 and 4x12. The program allows for any figure and does not support traps. The main reason for this is because insulation comes in a variety of square footages per roll.

Framing hardware, nuts, bolts, screws and mouldings are generally sold in units of 100, (C). Most everything else is sold by single units (EA). With these five calculations identified we may now start the bid.

The program as written may be used directly by lumbermen and contractors to make or check bids. A bid usually consists of two parts, an estimate of materials to be used and a cost quote that will be guaranteed for a limited period of time. LUMBER assumes the estimate of materials has been made.

PROGRAM OPERATION

LUMBER starts by requesting the name, address and job location of the customer. This section may be defeated by hitting RETURN if desired. The first choices are to STOP the input which then prints the bid, TAX which allows a percentage of sales tax and amount of dray to be entered or hit RETURN.

Hitting RETURN allows you to select board footage (BF), board footage calculated from lineal footage (LF), square footage (SF), price per C (C) or each (EA). Failing to enter the correct code will cause the sequence to be repeated.

After the printout the question CONTINUE is asked. This means should the program continue entering more information. Answering "N" concludes the program.

The method of inputting data was selected to provide the user with flexibility. Any number up to 35 calculations, tax or dray amounts may be entered, examined and re-entered prior to making the final formatted output.

Five questions are asked during the calculations:

1. NUMBER OF P.C. requests the actual full count which should be entered as a whole number.
2. SIDE 1, SIDE 2 requests the dimensions of each side of lumber separated by a comma. For example, a 2x4 is entered 2,4.
3. LENGTH requests the proper even length.
4. DESCRIPTION is a string that will identify the item. For example, you may wish to enter an item as 2x4x10 #2 HF S4S.
5. ENTER PRICE PER M requests the price per 1000 board or square feet. Lesser quality lumber ranges around $200/M while clear lumber may be around $2500/M.

The program's output includes a column for a line number, the number of pieces, the description, the calculated footage and per unit basis, the price per unit basis and the total cost for the given item. Care should be taken so as to not confuse unit basis for a dissimilar unit price. To calculate the 40 2x4 #2 studs at $1.89 remember to select EA and not BF.

This program was written for actual construction applications. With minor alterations it should prove useful under different types of lumber and building material applications. The straightforward program layout may be easily modified to any number of specific uses.☐

Program Listing Follows

*The price per M is $22.35 cheaper, but don't always assume that the price per M is cheaper. This program puts you on the same level as a lumberyard so you can know for sure.

PROGRAM LISTING

```
0010 REM LUMBER AND MATERIAL BID V 2.0
0040 FOR X9=1TO3:PRINTCHR$(26):NEXTX9:PRINTCHR$(2)
0050 DIM H$(35),G$(35),I(35),J(35),K(35),L(35)
0060 DEF FNA(J)=LEN(STR$(INT(J)))
0070 LINE= 120:STRING=28:DIGITS=2:PRINT
0080 INPUT "MATERIAL ESTIMATE FOR WHOM",A$
0090 INPUT "ADDRESS",B$
0100 INPUT "CITY, STATE AND ZIP",C$
0110 INPUT "JOB ADDRESS",D$
0120 INPUT "ESTIMATE BY: ",E$
0130 INPUT "DATE",F$
0140 INPUT "ENTER [STOP] [TAX] OR HIT RETURN ",X$
0150 IF X$="STOP" GOTO 250
0160 IF X$="TAX" GOTO 1770
0170 INPUT "UNIT: BF, LF, SF, C, EA ",G$
0180 IF G$="BF" THEN G=G+1:G$(G)=G$:GOTO 800
0190 IF G$="LF" THEN G=G+1:G$="BF":G$(G)=G$:GOTO 1120
0200 IF G$="SF" THEN G=G+1:G$(G)=G$:GOTO 1350
0210 IF G$="C" THEN G=G+1:G$(G)=G$:GOTO 1510
0220 IF G$="EA" THEN G=G+1:G$(G)=G$:GOTO 1640
0230 PRINT TAB(5);"> INPUT ERROR.  PLEASE RE-ENTER"
0240 GOTO 140
0250 FOR X9=1TO3:PRINTCHR$(26):NEXTX9:PRINTCHR$(2)
0260 PRINT :PRINT
0270 PRINT TAB(15);"JACK'S LUMBER AND MATERIALS"
0280 PRINT TAB(18);"ROUTE 1  BOX 123456"
0290 PRINT TAB(17);"CANYON, ARIZONA 85989"
0310 PRINT :PRINT
0320 PRINT TAB(13);"MATERIAL ESTIMATE AND PRICE QUOTE"
0330 FOR X9=1TO33:PRINTTAB(13);"-";:NEXTX9
0340 PRINT
0350 PRINT "NAME: ";TAB(10);A$;TAB(45);F$
0360 PRINT "ADDRESS: ";B$
0370 PRINT TAB(10);C$
0390 PRINT "JOB: ";D$;TAB(45);"EST. BY: ";E$
0400 PRINT :PRINT
0410 PRINT "LINE";TAB(6);"AMT";TAB(15);
0420 PRINT "DESCRIPTION";TAB(38);"FT/UNIT";TAB(49);"PRICE";
0430 PRINT TAB(57);"TOTAL"
0440 FOR X9=1TO64:PRINT"-";:NEXTX9:PRINT
0450 FOR X=1TOG
0460 DIGITS= 0
0470 PRINT X;
0480 PRINT TAB(9-FNA(I(X)));I(X);TAB(10);H$(X);
0490 IF J(X)=0 THEN 510
0950 LET J=INT(J+.5)
0960 LET J(G)=J
0970 INPUT "ENTER PRICE PER M ",N6
0980 LET K(G)=N6
0990 LET L=(N6/1000)*(J)
1000 LET L=INT(L*100+.5)/100
1010 LET L(G)=L
1020 LET T=T+L(G)
1030 GOTO 140
1040 PRINT TAB(5);"> INPUT ERROR.  YOU HAVE ENTERED AN UNCOMMON"
1050 INPUT "SIZE.  PLEASE VERIFY SIZE, [Y] OR [N]. ",X$
1060 IF X$<>"Y" THEN 810
1070 RETURN
1080 PRINT TAB(5);"> INPUT ERROR.  YOU HAVE ENTERED AN UNCOMMON"
1090 INPUT "LENGTH.  PLEASE VERIFY LENGTH, [Y] OR [N]. ",X$
1100 IF X$<>"Y" THEN 810
1110 RETURN
1120 REM BOARD FOOTAGE CACULATION THROUGH LINEAL FOOTAGE
1130 PRINT
1140 INPUT "LINEAL FEET ",O3
1150 LET I(G)=O3
1160 INPUT "SIDE 1, SIDE 2 ",O1,O2
1170 LET O5=O1*O2/12
1180 IF O5<.166666 GOSUB 1310
1190 IF O5>8.0 GOSUB 1310
1200 INPUT "DESCRIPTION ",H$
1210 LET H$(G)=H$
1220 INPUT "PRICE PER M ",K
1230 LET K(G)=K
1240 LET J=INT(O3*O5+.5)
1250 LET J(G)=J
1260 LET L=(K/1000)*J
1270 LET L=INT(L*100+.5)/100
1280 LET L(G)=L
1290 LET T=T+L(G)
1300 GOTO 140
1310 PRINT TAB(5);"> INPUT ERROR.  SIDE 1 AND SIDE 2 ARE"
1320 INPUT "UNUSUAL.  PLEASE VERIFY [Y] OR [N]. ",X$
1330 IF X$<>"Y" THEN 1130
1340 RETURN
1350 REM SQUARE FOOTAGE
1360 PRINT
1370 INPUT "NUMBER ",P3
1380 LET I(G)=P3
1390 INPUT "SIDE 1, SIDE 2 ",P1,P2
```

```
0500 PRINT TAB(42-FNA(J(X)));J(X);TAB(43);"/";G$(X); :GOTO 520
0510 PRINT TAB(43);"/";G$(X);
0520 DIGITS= 2
0530 PRINT TAB(55-FNA(K(X)));K(X);TAB(63-FNA(L(X)));L(X)
0540 NEXT X
0550 PRINT
0560 PRINT TAB(41);"DRAY ";TAB(63-FNA(T2));T2
0570 PRINT TAB(41);"TOTAL ";TAB(63-FNA(T));T
0580 LET M1=(M*T)/100
0590 LET M1=INT(M1*100+.5)/100
0600 DIGITS= 0:PRINTTAB(41);"TAX ";M;"%";
0610 DIGITS= 2:PRINTTAB(63-FNA(M1));M1
0620 PRINT TAB(41);"TOTAL";TAB(55);"$";TAB(63-FNA(M1+T));M1+T
0630 PRINT
0640 INPUT "CONTINUE ",X$
0650 IF X$="Y" THEN 140
0660 PRINT :PRINT
0670 INPUT "SUBJECT TO REVIEW AFTER ",W$
0680 INPUT "BEGIN SHIPPMENT BY ",U$
0690 INPUT "COMPLETE SHIPPMENT BY ",V$
0700 PRINT TAB(3);"INPUT ERRORS ARE SUBJECT TO CORRECTION.  IF OUR"
0710 PRINT "ESTIMATORS HAVE ARRIVED AT THE QUANITY OF MATERIAL SHOWN"
0720 PRINT "THEY HAVE MADE THE ESTIMATE TO THE BEST OF THEIR ABILITY.
0730 PRINT "JACK'S LUMBER CANNOT GUARANTEE THAT THE MATERIAL ESTIMATE'
0740 PRINT "WILL FULFULL YOUR REQUIREMENTS COMPLETELY.  JACKS'S LUMBER'
0750 PRINT "AGREES TO FURNISH MATERIAL LISTED AT THE UNIT PRICE"
0760 PRINT "SHOWN IF SHIPPMENT IS BEGUN BY ";U$;" AND COMPLETED "
0770 PRINT "BY ";V$;".  QUOTED PRICES ARE SUBJECT TO REVIEW "
0780 PRINT "AFTER ";W$;"."
0790 END
0800 REM BOARD FOOTAGE CACULATION
0810 PRINT
0820 INPUT "NUMBER OF PC.  ",N3
0830 LET I(G)=N3
0840 INPUT "SIDE 1, SIDE 2 ",N1,N2
0850 INPUT "LENGTH ",N4
0860 IF N4/2<>INT(N4/2) GOSUB 1080
0870 IF N4<6 GOSUB 1080
0880 IF N4>24 GOSUB 1080
0890 LET N5=N1*N2/12
0900 IF N5<.1666666 GOSUB 1040
0910 IF N5>8.0 GOSUB 1040
0920 INPUT "DESCRIPTION ",H$
0930 LET H$(G)=H$
0940 LET J=(N3*N4)*N5
```

```
1400 INPUT "DESCRIPTION ",H$
1410 LET H$(G)=H$
1420 LET J=INT((P1*P2)*P3+.5)
1430 LET J(G)=J
1440 INPUT "PRICE PER M ",P6
1450 LET K(G)=P6
1460 LET L=(P6/1000)*(J)
1470 LET L=INT(L*100+.5)/100
1480 LET L(G)=L
1490 LET T=T+L(G)
1500 GOTO 140
1510 REM PER 100
1520 PRINT
1530 INPUT "TOTAL ",R1
1540 LET I(G)=R1
1550 INPUT "DESCRIPTION ",H$
1560 LET H$(G)=H$
1570 INPUT "PRICE PER C ",K
1580 LET K(G)=K
1590 LET L=(K/100)*(R1)
1600 LET L=INT(L*100+.5)/100
1610 LET L(G)=L
1620 LET T=T+L(G)
1630 GOTO 140
1640 REM PER EACH
1650 PRINT
1660 INPUT "QUANITY ",I
1670 LET I(G)=I
1680 INPUT "DESCRIPTION ",H$
1690 LET H$(G)=H$
1700 INPUT "UNIT PRICE ",K
1710 LET K(G)=K
1720 LET S=I*K
1730 LET S=INT(S*100+.5)/100
1740 LET L(G)=S
1750 LET T=T+L(G)
1760 GOTO 140
1770 REM ENTER TAX & DRAY
1780 PRINT
1790 INPUT "ENTER % OF SALES TAX ",M
1800 INPUT "DRAY: ",T2
1810 LET T=T+T2
1820 GOTO 250
1830 END
```

COMING NEXT MONTH

As schools across the country close for the summer, INTERFACE AGE picks up some of the slack with a new monthly feature, The Learning Center. This section of the magazine will consist of a variety of educational functions of micros, whether the student in the world of computing is a novice or an old hand. Subjects will range from scholastic uses to learning how the computer works to July's feature on using the micro to communicate with dolphins.

The who, what and why of OEMs will also be examined next month. Along with the Original Equipment Manufacturers, the makers of computer system furniture will be investigated.

And the newest of the INTERFACE AGE tutorials will begin. The Pascal Notebook, by Associate Editor Henry Davis, starts with some of the history and theory of this powerful new programming language.

And last but not least, there are the usual columns, hardware and software features that make INTERFACE AGE the magazine for the microcomputing enthusiast.

For your SWTP 6800 Computer . . .

PERCOM's™ FLOPPY DISK SYSTEM the LFD-400

Ready to plug in and run the moment you receive it. Nothing else to buy, no extra memory. No "booting" with PerCom MINIDOS™, the remarkable disk operating system on EPROM. Expandable to either two or three drives. Outstanding operating, utility and application programs.

For more information see your local PerCom dealer or call toll free 1-800-527-1592

only **$599.95**

fully assembled and tested shipping paid

PERCOM™

PERCOM DATA COMPANY, INC.
Dept. I 211 N. Kirby Garland, Texas 75042
(214) 272-3421

For the low $599.95 price, you not only get the disk drive, drive power supply, SS-50 bus controller/interface card, and MINIDOS™, you also receive:

• an attractive metal enclosure • a fully assembled and tested interconnecting cable • a 70-page instruction manual that includes operating instructions, schematics, service procedures and a complete listing of MINIDOS™ • technical memo updates — helpful hints which supplement the manual instructions • a 90-day limited warranty.

SOFTWARE FOR THE LFD-400 SYSTEM

Disk operating and file management systems

INDEX™ The most advanced disk operating and file management system available for the 6800. INterrupt Driven EXecutive operating system features file-and-device-independent, queue-buffered character stream I/O. Linked-file disk architecture, with automatic file creation and allocation for ASCII and binary files, supports sequential and semi-random access disk files. Multi-level file name directory includes name, extension, version, protection and date. Requires 8K RAM at $A000. Diskette includes numerous utilities $99.95

MINIDOS-PLUSX An easy-to-use DOS for the small computing system. Supports up to 31 named files. Available on ROM or diskette complete with source listing $39.95

BASIC Interpreters and Compilers

SUPER BASIC A 10K extended disk BASIC interpreter for the 6800. Faster than SWTP BASIC. Handles data files. Programs may be prepared using a text editor described below $49.95

BASIC BANDAID™ Turn SWTP 8K BASIC into a random access data file disk BASIC. Includes many speed improvements, and program disk CHAINing $17.95

STRUBAL+™ A STRUctured BAsic Language compiler for the professional programmer. 14-digit floating point, strings, scientific functions, 2-dimensional arrays. Requires 16K RAM and Linkage Editor (see below). Use one of the following text editors to prepare programs. Complete with RUN-TIME and FLOATING POINT packages $249.95

Text Editors and Processors

EDIT68 Hemenway Associates' powerful disk-based text editor. May be used to create programs and data files. Supports MACROS which perform complex, repetitive editing functions. Permits text files larger than available RAM to be created and edited $39.95

TOUCHUP™ Modifies TSC's Text Editor and Text Processor for PerCom disk operation. ROLL function permits text files larger than available RAM to be created and edited. Supplied on diskette complete with source listing $17.95

Assemblers

PerCom 6800 SYMBOLIC ASSEMBLER Specify assembly options at time of assembly with this symbolic assembler. Source listing on diskette $29.95

MACRO-RELOCATING ASSEMBLER Hemenway Associates' assembler for the programming professional. Generates relocatable linking object code. Supports MACROS. Permits conditional assembly $79.95

LINKAGE EDITOR — for STRUBAL+™ and the MACRO-Relocating assembler $49.95

CROSS REFERENCE Utility program that produces a cross-reference listing of an input source listing file $29.95

Business Applications

GENERAL LEDGER SYSTEM Accommodates up to 250 accounts. Financial information immediately available — no sorting required. Audit trail information permits tracking from GL record data back to source document. User defines account numbers $199.95

FULL FUNCTION MAILING LIST 700 addresses per diskette. Powerful search, sort, create and update capability $99.95

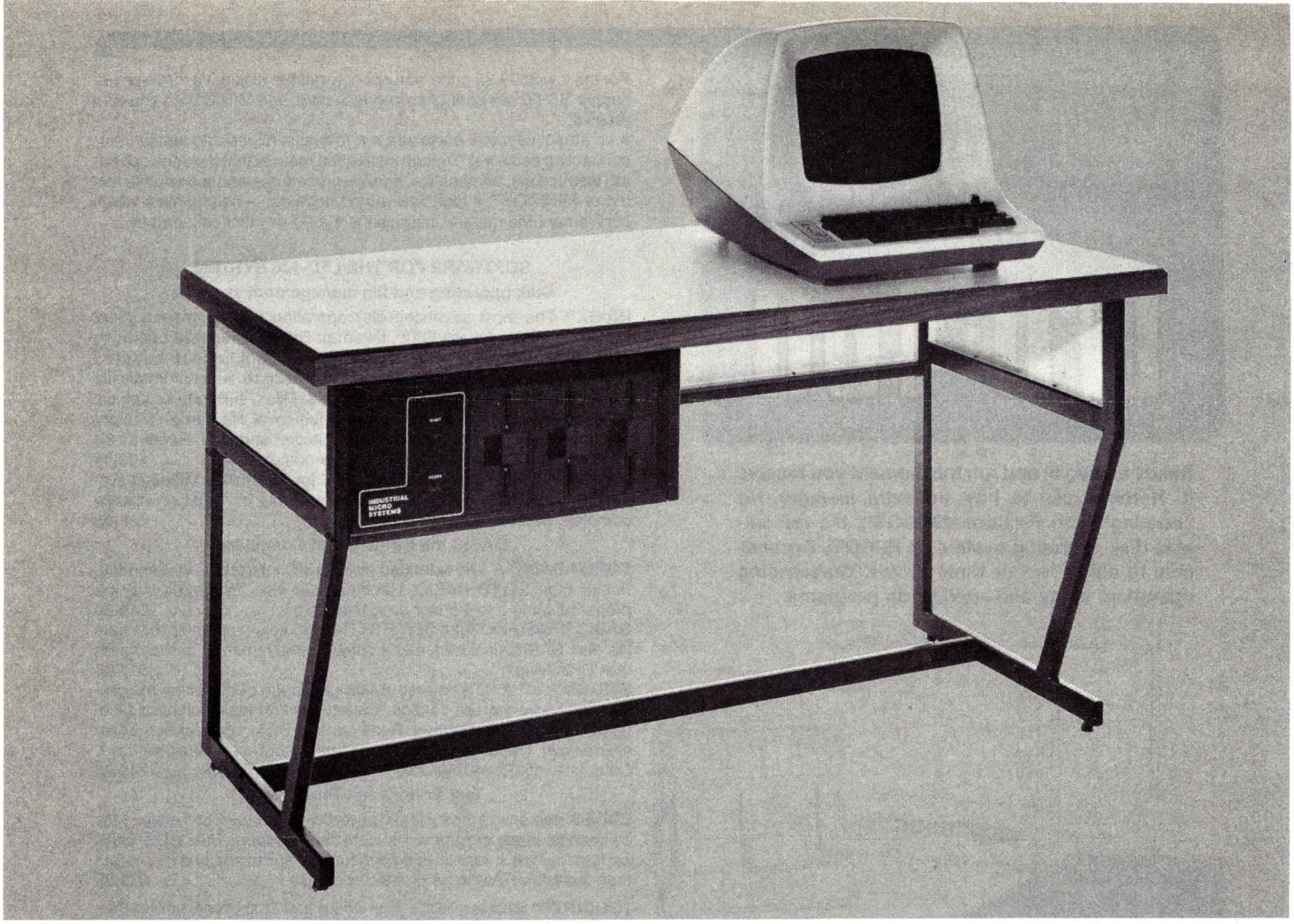
PERCOM FINDER™ General purpose information retrieval system and data base manager $99.95

™ trademark of PERCOM Data Company, Inc.

Ordering information

To order, call toll free *1-800-527-1592.* MC and VISA welcome. COD orders require 30% deposit plus 5% handling charge. Allow three weeks for delivery. Allow three extra weeks if payment is by personal check. Texas residents add 5% sales tax.

PERCOM ***'peripherals for personal computing'***


INDUSTRIAL
MICRO
SYSTEMS

Interfacing A Numerical Processor Chip to the TRS-80

By James E. Randall

The advantage of numerical processing chips is that they can perform number crunching rapidly using minimal memory. Such a feature may not seem important to many TRS-80 users because this microcomputer comes with software math routines in the Level II BASIC on a ROM and these operate at speeds sufficient for most games and business applications. However, the widespread availability of this micro and the expected operating system support for more sophisticated applications suggests that there may be a significant number of scientific computations which could benefit from this numerical enhancement in a low-cost system.

The concept presented here is to retain the ROM BASIC for I/O and formatting and then use the USR, PEEK and POKE commands to communicate with machine language routines which utilize the features of the numerical processor. As an example, the author uses microcomputers for teaching-exercises in physiology, including graphic simulations of the nerve action potential. To do this in high-level language required about two minutes for 256 computed points on the graph, a time so long that it discouraged student participation. However, this time was reduced to a very reasonable ten seconds by using BASIC to ask for stimulus parameters such as amplitude and duration, then transferring these values to a machine language routine which did the computations and plotting. This latter routine called the arithmetic processor functions from a set of short servicing subroutines in a logical sequence that paralleled what would have been written in BASIC commands.

The AM-9511, made by Advanced Micro Devices, interfaces easily with the Z-80 through the TRS-80 expansion-port edge connector and requires only a power supply, 2-MHz clock, and one decoding circuit to be implemented. In addition, this arithmetic processing unit (APU) uses a binary 4-byte floating point format that is very similar to that used in Level II BASIC, making it particularly easy to pass arguments between the two kinds of programs. This article describes the interfacing circuit, compares the two floating point formats involved, and gives a representative APU routine. For simplicity and clarity the treatment will not include the error flags nor will it discuss interrupt information which is available from the APU. The reader should acquire detailed specifications and algorithm details from the manufacturer. This chip currently costs $200 in unit quantities but it approaches the speed and accuracy of $10,000 floating-point processors used with minicomputers a decade ago.

THE AM-9511 NUMERICAL PROCESSOR CHIP

This 24-pin chip requires +5 and +12 volts at about 100 ma each and communicates with the TRS-80 through 8 bi-directional data bus pins. The remaining pins accept control signals to determine the direction of information transfer and also provide signals which could activate an interrupt when a computation has been completed. Data, commands, and error information are transferred in 8-bit bytes between the data bus lines available on the TRS-80 rear connector and the data pins of the APU chips. I/O port commands from the Z-80 microprocessor in the TRS-80 can be used to execute these transfers. When a data output command is issued, the byte in the accumulator is pushed onto the top of a stack of 16 8-bit registers within the APU. Conversely, a data input command pops the top byte from the stack and transfers it to the accumulator. An appropriate output command from the Z-80 initiates the math operation coded in the accumulator using the stack bytes as operands.

The arithmetic operations treat the bytes on the stack in one of three different ways: single precision integer math involves two successive bytes as a 16-bit integer; double precision using four bytes as a 32-bit integer; and floating point using four bytes, one for the exponent and the signs of exponent and of mantissa, the remaining three bytes form a 24-bit mantissa normalized for a minimum value of 0.5. The operations themselves include the four functions, a wide range of transcendental functions, and also means for conversion between the three different forms. In use, the computer would push the bytes for operand(s) onto the stack; give the code for the desired math operation; test the status bit until the computation is completed; and then pop the result from the stack for use by the computer itself. The overall efficiency is increased if the operands for successive math operations can be left on the APU stack thus reducing the time required for pushing and popping and memory transfers.

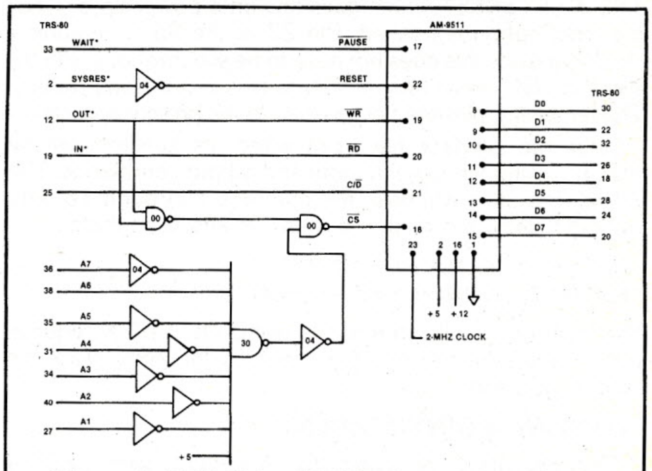


Figure 1. TRS-80 to AM-9511 Interconnections.

Figure 1 gives the decoding circuit and interconnections required to connect the AM-9511 to the TRS-80 interface connector. The APU CHIP SELECT (pin 18) is enabled by decoding seven of the address lines (A1-A7) according to the port number being used, and also by either a Z-80 INPUT or OUTPUT command as signaled by the TRS-80 IN* or OUT* going low. The port number which would be selected in the schematic would enable the APU chip for the Z-80 commands:

IN A,(64)   IN A,(65)   OUT (64),A   OUT (65),A

Note that for testing purposes these could be implemented by BASIC commands, though the overhead would defeat any timing advantages.

The A0 line, connected directly to the APU pin 21, determines whether the byte transferred on the data bus is a command or data. Odd port numbers (65 in this case) have this

pin high for Z-80 input and output commands and the data bus is treated as either an opcode or status information. Even port numbers, such as 64, indicate that data is to be either pushed onto or popped off of the APU internal stack. Two other pins determine the direction of these command/data transfers. When the Z-80 issues an output the connector pin OUT* goes low which places the APU WRITE (pin 19) low and the computer accumulator goes to the APU, either to start a specific math operation or to push the byte onto the stack. When an input command is issued the TRS-80 IN* signal brings the APU READ (pin 20) low and this sees that the byte transfer is from the APU to accumulator, either status and error information or a data byte popped from the stack. Table 1 outlines the different possible combinations of these signals. The author found that data, address, and control lines were sufficiently buffered internally in the TRS-80 for direct connection to the APU and also to a UART used to drive a printer.

FUNCTION	LEVELS			ACTIVATED BY	
AM-9511	C/D	READ	WRITE	Z-80	BASIC
TRS-80	A0	IN*	OUT*		
COMMAND	HI	HI	LO	OUT (65),A	OUT 65,C
STATUS, ERRORS	HI	LO	HI	IN A,(65)	S = INP(65)
PUSH DATA	LO	HI	LO	OUT (64),A	OUT 64,D
POP DATA	LO	LO	HI	IN A,(64)	D = INP(64)

Table 1. Numerical Processor Control Signals

Another signal, PAUSE on APU pin 17, is connected directly to the TRS-80 WAIT* terminal. This allows the APU to place the Z-80 in a wait state until information has been transferred from the internal registers to the data pins; when the pause goes high the Z-80 continues and places the byte into the accumulator. The APU RESET (pin 22) should be pulled high to initialize that chip. This can be done by inverting the TRS-80 SYSRES* which goes low on a power up or when the reset button is pushed. Pin 23 of the 9511 requires a clock but since this does not have to be synchronous with the Z-80 any RC circuit near 2 MHz seems to function properly. The remaining pins are not involved in the present application.

Once the interface has been wired, its function can be checked using the Level II input and output commands. The following routine will push the numbers 1 through 16 onto the APU stack and then pop them off and print them.

```
FOR I=I TO 16; OUT 64,I:  NEXT I
FOR I=I TO 16; PRINT INP(64);:  NEXT I.
```

The testing of math commands using OUT 65,X requires some knowledge of how the bytes on the APU stack are used as operands.

FLOATING-POINT FORMAT

One advantage of using the AM-9511 with Level II BASIC is that the two use very similar 4-byte formats for binary floating-point computations. Two of the bytes are identical for the two systems and the other two require a reordering of those bits which indicate the signs of the mantissa and exponents. The author found the most expedient method of communication between BASIC and APU routines was to poke the four bytes located by VARPTR(X) into available memory where a machine language subroutine could reshuffle them into the format for the APU computations. The final result of floating-point computations could then be placed in BASIC format where PEEK and POKE commands could move them into previously located BASIC variables.

Table 2 illustrates the two floating point formats. In both cases one byte conveys the exponent (power of two) and three bytes represent a 24-bit mantissa with the most significant bit representing decimal 0.5 and always being considered as set unless the variable's value is zero. For the BASIC case this most significant bit is implied and the explicit bit position determines the sign of the mantissa; the remaining 23 bits tell the value of the fraction to be added to 0.5. If the BASIC variable is X, the contents of VARPTR(X) gives the least significant byte of the mantissa; those of location VARPTR(X)+1 give the middle byte.

```
BASIC FORMAT

    VARPTR(X)+3     VARPTR(X)+2     VARPTR(X)+1     VARPTR(X)
     00000000       0 0000000        00000000        00000000
                    ↑         23 bits used as 24-bit mantissa
     Exponent       |         Most sig. bit always set, but implied.
   (Offset 128)     |
                    This bit set for negative mantissa

AM-9511 FORMAT

    Top of Stack    Next down       Third down      Fourth down
    0 0000000       10000000         00000000        00000000
    ↑  Exponent               24-bit mantissa
    |  (Offset 128)           Most sig. bit set expect for X=0
    Bit set for
      negative mantissa
```

Table 2. Four-byte Floating-point formats for AM-9511 and Microsoft BASIC.

The next location contains the most significant mantissa byte except that bit 7 in the byte is set for a negative mantissa and there is no explicit bit for the 0.5 part of the mantissa. If location VARPTR(X)+3 has a value of 0, the variable is 0; otherwise the byte tells the exponent of two to be applied to the fractional mantissa. Positive and negative exponents in the range -127 to $+127$ are covered by having this fourth byte expressed with an offset of 128; values greater than this being considered positive, values less than this being negative exponents.

Execution of the following routine can offer insight to this format:

```
10 INPUT "X = "; X
20 FOR I = 0 TO 3: PRINT PEEK(VARPTR(X)+I): NEXT I
30 GOTO 10
```

When running this exercise, note how the third and fourth bytes respond to changes in sign of the exponent and of the mantissa.

When a four-byte floating-point operand is pushed onto the AM-9511 stack, the first byte pushed is the least significant one (the same as that at VARPTR(X) in BASIC); the second one pushed is the middle byte from VARPTR(X)+1. The third one pushed is the most significant mantissa byte except that the convention used always has bit 7 set explicitly unless the variable is a 0. In the APU convention the mantissa sign appears as bit 7 on the fourth byte to be pushed onto the stack. The other bits (0-6) of this last byte give the exponent expressed as an offset of 128 except that positive exponents are in only the low 6 bit locations. For example, if the byte were equal to 1, this would represent a positive mantissa to be multiplied by $+2$; if the byte were 129, a negative mantissa multiplied by $+2$; if 127, a positive mantissa multiplied by 2^{-1}; if 255, a negative mantissa times 2^{-1}.

The following BASIC routine can be used to generate floating-point numbers on the AM-9511 stack from which they can be read as an illustrative exercise.

```
10 INPUT "POSITIVE INTEGER = "; X
20 OUT 64,(XAND255): OUT 64,INT(X/256)
30 OUT 65,29            'FLOATS INTEGER
40 IF (INP(65)AND128)=128 THEN GOTO 40
50 FOR I=0 TO 3: PRINT INP(64): NEXT I
60 GOTO 10
```

Line 20 pushes a positive integer onto the APU stack as two 8-bit bytes; Line 30 commands the APU to convert these to floating point format; line 40 is repeated until the APU status bit goes low; and then the stack is popped off and printed. Negative integers can be acquired by adding the APU operation of changing the floating point operand's sign (OUT 65,21); fractional values can be formed from division of inte-

gers to generate negative exponents.

Conversion between the two floating-point formats can most quickly be done in machine language but the general principle can be illustrated by the following BASIC routine which pushes a BASIC variable onto the APU stack in correct 4-byte format and then pops and prints them for examination.

```
 10 INPUT "ANY VARIABLE = "; X
 20 IF PEEK(VARPTR(X)+3)=0 THEN
      FOR I=0to3: OUT 64,0:NEXT I: GOTO 100
 30 OUT 64, PEEK(VARPTR(X))          'PUSH LEAST
 40 OUT 64, PEEK(VARPTR(X)+1)        'PUSH MID
 50 OUT 64,(PEEK(VARPTR(X)+2)OR128)     'MOST
 60 'IF LAST BYTE => 128 THEN POSITIVE EXP
 70 Z=PEEK(VARPTR(X)+3): IF Z=>128 THEN
      OUT 64,(ZAND 63)+(PEEK(VARPTR(X)+2)AND128):
      GOTO 100
 80 'IF LAST BYTE <128 THEN NEG EXP; USE 7 BITS
 90 OUT 64,Z+(PEEK(VARPTR(X)+2)AND128)
100 FOR I=0 TO 3: PRINT INP(64): NEXT I
```

Lines 30 and 40 push the least and middle significant bytes without modification. The APU format requires that the most significant byte of the mantissa always have the high bit set, this is done in line 50. Line 70 tests the BASIC exponent byte, if the high bit is set it keeps the mantissa sign and the low 6 bits of the exponent. The negative exponent, handled by line 90, keeps the low 7 bits and picks up the mantissa sign.

MACHINE-LANGUAGE PROGRAMMING

The format conversion subroutines written in machine language occupy about 30 bytes each and are entered with one register pair (HL) set to the memory location of the source format and another pair (DE) set to the location of the designation format. These subroutines will not be given here but they involve transferring the two least significant bytes unchanged and setting the correct sign bits in the other two.

The following machine-language routine could be called by the BASIC USR(0) function with the BASIC variable X already poked in the four locations starting at 32256. This routine converts the variable into APU format, pushed it onto the stack, does a sequence of math operations, pops the result, converts it to BASIC format, then returns to BASIC. The BASIC routine would pick up the results and, perhaps, print them. The greatest advantage of the numerical processor chip is realized when the math operations are extensive and the shuffling to BASIC is minimal.

```
100 ;Z-80 MACHINE LANGUAGE
110 BASVAR  EQU    32256           ;BASIC FORM
120 APUVAR  EQU    32260           ;APU VARIABLE
130 ;CONVERT BASIC TO APU FORMAT
140         LD     (HL),BASVAR     ;FROM
150         LD     (DE),APUVAR     ;INTO
160         CALL   BASAPU          ;CONVERT
170 ;PUSH          ONTO            APU
180         LD     (HL),APUVAR     ;POINTER
190         CALL   PSHFLT          ;PUSH
200 ;DO MATH OPERATIONS AS REQUIRED
210         CALL   EXP             ;EXP(X)
. . .
400 ;NOW POP OFF STACK INTO MEMORY
410         LD     (HL),APUVAR     ;POINTER
420         CALL   POPFLT          ;POP TO MEM
430 ;CONVERT TO BASIC FORMAT
440         LD     (HL),APUVAR     ;FROM
450         LD     (DE),BASVAR     ;INTO
460         CALL   APUBAS          ;CONVERT
470 ;RETURN TO BASIC
480         RET
490 ;MATH SERVICING ROUTINE EXAMPLE
500 EXP     CALL   BUSY            ;APU DONE?
510         LD     A,16            ;OP CODE
520         OUT    (65),A          ;EXP(X)
530         RET
540 ;CHECKS APU STATUS BIT
550 BUSY    IN     A,(65)          ;STATUS,ERRORS
560         AND    128             ;HI BIT
570         JR     NZ,BUSY         ;LOOP TILL LOW
580         RET
590         END
```

Figure 2.

Such machine language programming is tedious to write and debug when there is only cassette tape for mass storage but no doubt disk editors and assemblers are in the offing for the TRS-80. In any case, if rapid number crunching is required, the hardware effort is minimal and the software development is a good investment. No doubt future versions of high-level languages will be designed to take full advantage of numerical processing chips.☐

System of the Month

The Sord Mark II Series

By Tom Fox, Systems Editor

One of the newest entrants into the fields of business and personal computers is a Kansas City-based firm called Sord U.S.A. This company, which is an extension of Sord Electronics of Japan, has begun offering their line of computers to the U.S. marketplace with a great deal of earnest during the past several months.

Although the first introduction of the system was in May of 1978, the firm did not begin to actively market it in the United States until late in 78. This was primarily due to problems in setting up an adequate system of importing the units, and establishing a dealer network.

During the last four months, Sord U.S.A. has permeated the computer field, which up until now has been dominated by U.S. firms. They have also been setting up contracts with outside software firms for the development of application software specifically geared to the Sord equipment. Their plans are to be offering a complete system with software soon. This is possible, according to company officials, due to the high technology of the system and the fairly rapid development of applications software for several specific businesses.

THE BASIC SYSTEM

PHOTO 1 M223 Mark II

At the top of the Sord line is the Model M223 business system (Photo 1). This system comes complete with two Micropolis dual density floppy disks, CRT, keyboard, and 64K bytes of memory. The system has a total memory capacity of 128K bytes which is set up for the user and system.

The system is designed around the popular S-100 bus and the backplane has four slots, one of which is used for the 64K memory board and the others free for future expansion such as A/D converters, OCR readers and separate interfaces. Even though the backplane may appear small in comparison to other S-100 bus systems, it is more than sufficient when all of the built-in options are considered.

These options include the keyboard, which provides all ASCII characters for input, including graphic characters and a numeric keypad plus special function or option keys. These keys allow the operator to set up special repetitive functions and leave the machine alone to perform the data processing work. These special function keys are something new that have been added and came about as a result of extensive research on what businessmen and particularly secretaries, wanted in a machine.

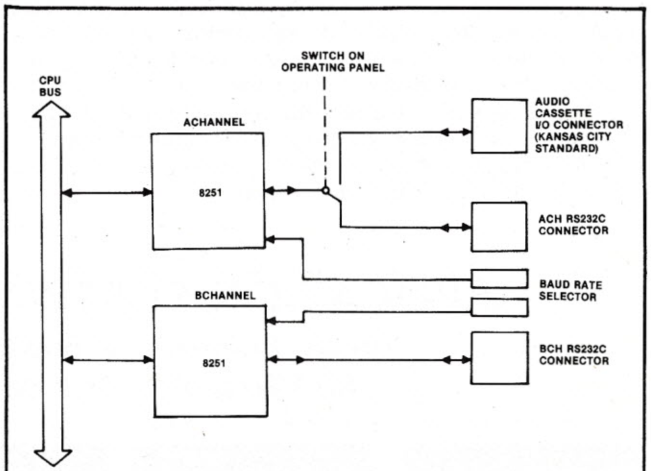
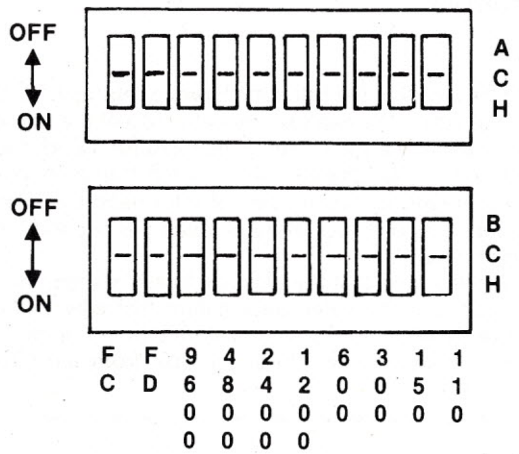


Figure 1.

The CRT, although an integral part of the system, can be considered a built-in option due to its flexibility. The CRT provides the ability to display up to 1920 characters in an 80 character by 24 line display. Complete graphics include moving graphics and reversible characters.

The built-in Micropolis mini-disks give up to 315K bytes of user storage and the advantage of being a well known and easily serviced brand.

Along with these options are two RS-232 ports and an audio cassette port. The ports are controlled from the front panel, allowing selection between the cassette or RS-232 port. The other serial port is for MODEM operation. In our test laboratory this was used both with a Lear Siegler ADM 1 terminal and a TELECOM 300 baud acoustic coupler to talk to the INTERFACE AGE answering system. The port for this experiment was handled completely under program control. Figure 1 shows an example of how the system is configured for the ports. Both serial ports maybe configured for separate baud rates by two DIP switches located on the main board. Figure 2 is an example of how these switches may be set up.



(SWITCHES LOCATED ON MAIN SYSTEM BOARD)

Figure 2.

SOFTWARE

Any computer system is only as good as the software. The software supplied with the system consists of an operating system that provides for total interactivity of the system plus the same high level of activity as a CP/M type system does. The operating system is not CP/M and currently no plans are set up to provide CP/M.

The Sord operating system can control up to four disk devices. The disk drives may be part of the built-in drives or add ons. Sord is currently working on a release to use hard disks and floppies at the same time.

With the operating system comes Sord extended BASIC plus. This BASIC is an interpreter but the company will be releasing a compiler version in the near future. The Sord BASIC does offer speed and higher accuracy than some of the currently available BASICs. Also scheduled to become available are: FORTRAN IV, COBOL and text processing software.

For the most part the software that is currently offered maximizes the power of the Z-80 processor and the 2.5 MHz operating speed of the system. Care has been taken to allow the software — BASIC — to communicate to even the slowest of peripheral devices without inhibiting the rest of the system.

COMING ATTRACTIONS

Because Sord has made a major commitment to service the growing business and personal computer markets, they have a number of developments underway or already available.

One of these new developments is the Model 203 personal computer which offers the same power as the M223, but in a lower cost range. This system (see Photo 2) is designed to fit the needs of the personal user and can be purchased with or without disks, but is readily expandable.

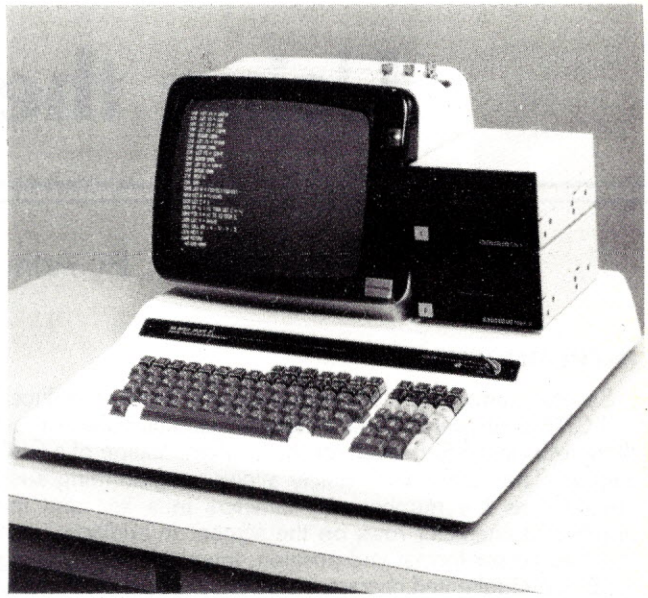


PHOTO 2 M203 Mark II

Along with the smaller system Sord is developing bus compatible PROM writers, real time clocks and in circuit emulators. According to Bob Chambers, vice president of marketing, there are a number of developments that will be available in the fourth quarter of this year, one of which is the availability of a hard disk system.

COSTS AND CONFIGURATIONS

The Sord systems are offered in a number of configurations dependent upon the user's needs, including application software requirements. Basically the M223 Business system with a printer sells for under $7,000 including system software. The M203 personal system sells for less although at press time the actual price ranges were not available.

The applications packages currently being developed range from real estate to general ledgers. Each application offers a unique approach in that the supplied documentation includes a C.A.I. (Computer Assisted Instruction) diskette to guide the user when first learning the system.

Because Sord is just now making a concentrated effort to reach the business market, they are interested in hearing from businessmen regarding specific applications. Sord may be contacted by calling or writing Sord U.S.A., 8300 NE Underground Drive, Kansas City, Missouri 64161, 1-800-821-5436.

DOCUMENTATION

The one area where Sord is weak is in their current documentation. However, a new systems manual and language manuals are under development by a U.S. firm. Sord is planning the issuance of the new manuals sometime by late summer or early fall. Even though the current system documentation manual does not come up to par with the rest of the system, it does cover all points of the system, plus offer many useful programming and hardware hints.□

CRT Monitor Design Using the Intel 8275

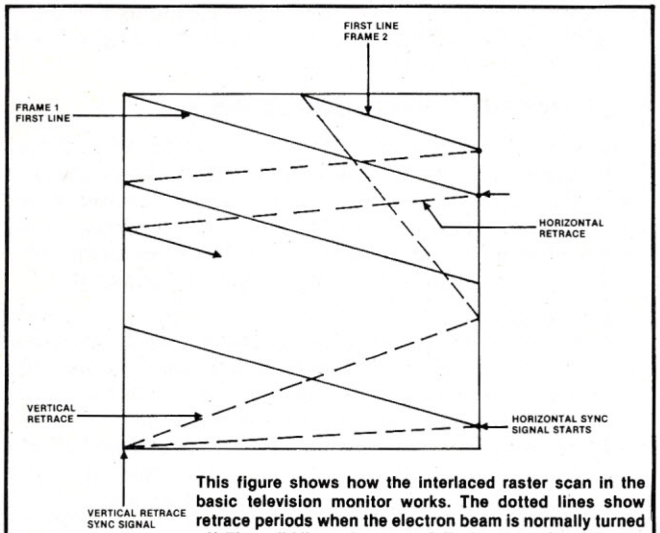
By Arthur A. Carapola

INTRODUCTION

Several semiconductor manufacturers have introduced LSI-CRT controllers. These integrated circuits replace most of the MSI and SSI needed for the implementation of a CRT display. Their extreme flexibility allows programming such functions as the number of characters in a line and the number of character rows on the screen. In addition, such items as cursor format and position, inverse video, blinking fields and highlighted characters can be provided with little or no additional hardware.

This article describes the Intel 8275 CRT controller, along with ideas on how to implement a sophisticated computer monitor using it.

RASTER SCAN CRT



This figure shows how the interlaced raster scan in the basic television monitor works. The dotted lines show retrace periods when the electron beam is normally turned off. The solid lines show actual display time when video information is being painted on the CRT screen.

Figure 1. A television interlaced raster scan.

Most people are familiar with the basic television raster scan technique (Figure 1). The picture is formed by sweeping the electron beam across the face of the picture tube. This sweep starts when the television receiver receives a horizontal sync pulse. During the retrace time, the electron beam is turned off (blanked). The time for one complete horizontal line is 63.5 microseconds. Of this, 53.5 microseconds is usable display time, the rest is for retrace. As the electron beam sweeps across the screen horizontally, the vertical sweep deflects it from the top of the screen to the bottom. The vertical sweep operates at 60 Hz giving 262.5 lines in one frame. The same way that the horizontal sweep returns to its starting point, the vertical sweep returns to the top of the frame. During vertical retrace the screen is blanked for 1250 microseconds leaving 242 usable lines in each frame. A complete picture consists of two frames of information — interlaced scanning. This provides thirty pictures per second and 484 lines per picture.

During the time that the electron beam is scanning across the screen, it is the video information that tells it whether a particular position on the screen should be on or off. This information is usually sent in serial form from the CRT controller board.

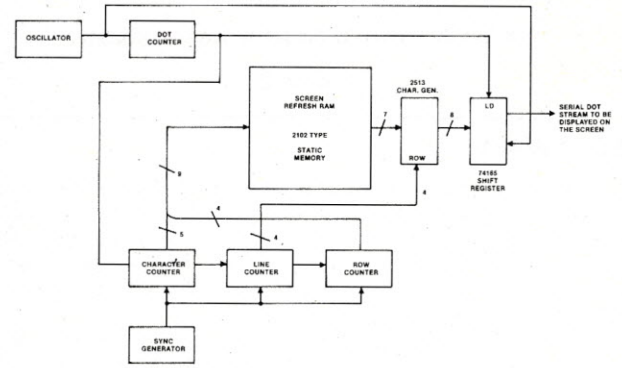


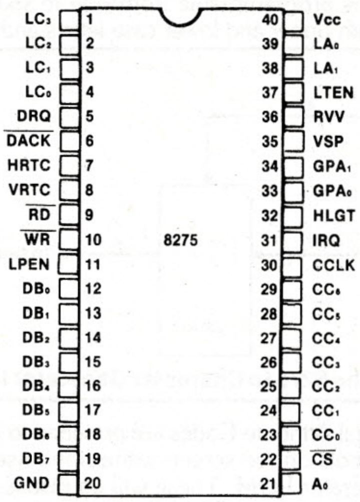
Figure 2. A typical raster scan refresh controller.

A block diagram of a typical raster scan television display controller is shown in Figure 2. When a character is accessed in the refresh RAM, it selects the appropriate dot pattern in the 2513 character generator. This pattern is then loaded into the 74165 8-bit shift register. The data is shifted out of the 74165 one bit at a time to be displayed on the TV screen. The function of the *Dot Counter* is to allow a precise number of clock cycles to go to the shift register. The example utilizes a ÷7 dot counter. Using a 5x7 character generator gives the 5 desired horizontal patterns plus two spaces.

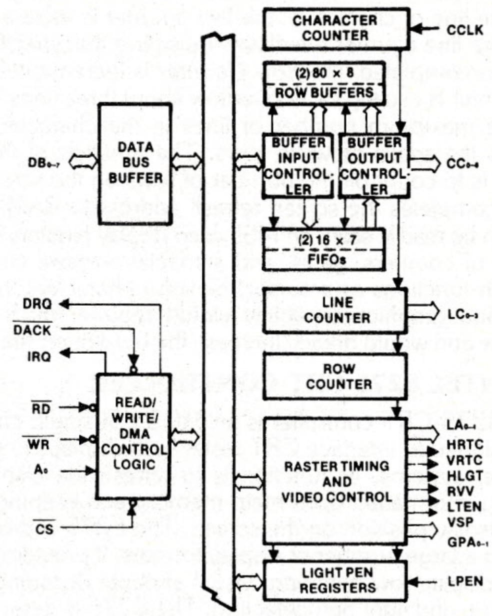
When the dot counter counts its specified number of clock cycles, it reloads the shift register and increments the *Character Counter.* The character counter counts the number of characters in the line. The outputs of the character counter are used as part of the refresh address to RAM. These outputs, along with the row counter outputs yet to be described, provide an exact coordinate in memory for the desired display position.

Due to the horizontal nature of the video sweep, a specific character must be considered as being composed of several

PIN CONFIGURATION

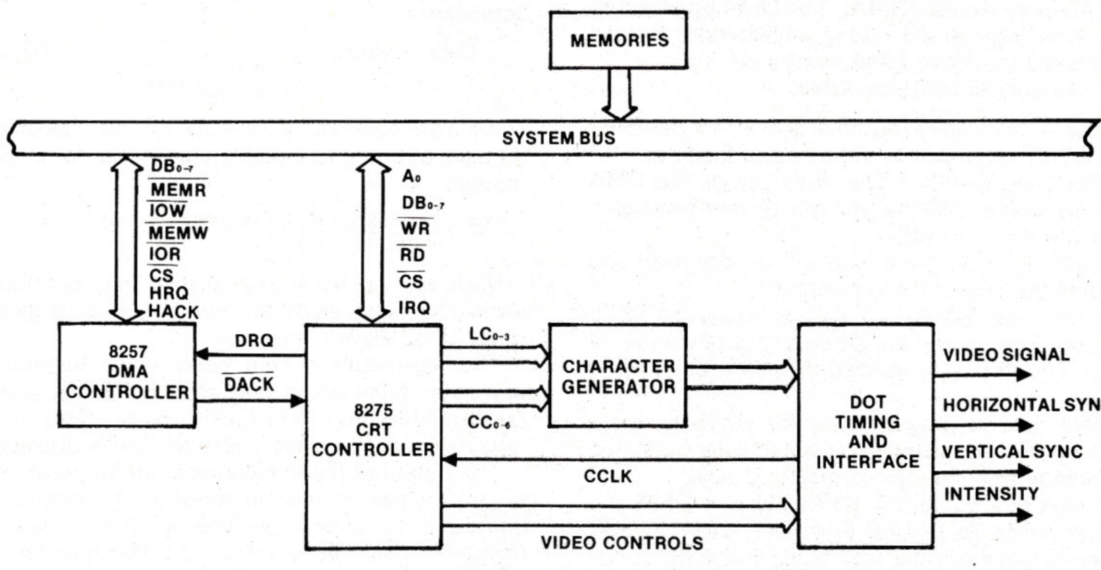


BLOCK DIAGRAM



PIN DESCRIPTION

DB_{0-7}	BI-DIRECTIONAL DATA BUS	LC_{0-3}	LINE COUNTER OUTPUTS
DRQ	DMA REQUEST OUTPUT	LA_{0-1}	LINE ATTRIBUTE OUTPUTS
$\overline{DACK}$	DMA ACKNOWLEDGE INPUT	HRTC	HORIZONTAL RETRACE OUTPUT
IRQ	INTERRUPT REQUEST OUTPUT	VRTC	VERTICAL RETRACE OUTPUT
$\overline{RD}$	READ STROBE INPUT	HLGT	HIGHLIGHT OUTPUT
$\overline{WR}$	WRITE STROBE INPUT	RVV	REVERSE VIDEO OUTPUT
A_0	REGISTER ADDRESS INPUT	LTEN	LIGHT ENABLE OUTPUT
CS	CHIP SELECT INPUT	VSP	VIDEO SUPPRESS OUTPUT
CCLK	CHARACTER CLOCK INPUT	GPA_{0-1}	GENERAL PURPOSE ATTRIBUTE OUTPUTS
CC_{0-6}	CHARACTER CODE OUTPUTS	LPEN	LIGHT PEN INPUT



8275 SYSTEMS BLOCK DIAGRAM

Figure 3.

horizontal lines of information. Therefore, along with the ASCII representation of the desired character, the character generator must also know what line of information is being displayed. This is the function of the *Line Counter.* The outputs of the line counter are sent to the character generator line input. Each time the character counter has completed a complete line of characters, the line counter is incremented. When the line counter overflows indicating that specific row has been completed, the *Row Counter* is incremented. The line counter is usually set to overflow about three lines higher than the maximum number of lines in the character. This gives us the space between rows. The function of the row counter is to count off the number of rows on the screen. Its output completes the screen refresh address to RAM.

As can be readily seen, an MSI video display requires a great amount of counters, gates, and a special purpose chip. To add such functions as a cursor, blinking characters, inverse video, and graphics capability would require much more hardware and would greatly increase the burden on the CPU.

THE INTEL 8275 CRT CONTROLLER

The 8275 CRT controller is an MOS-LSI single chip device designed to interface CRT raster scan displays to microcomputer systems. Its function is to refresh the display by buffering information from main memory and keeping track of the display position on the screen. The 8275 is programmable to a large number of display formats. It provides raster timing, display row buffering, visual attribute decoding, cursor timing, and light pen detecting. The 8275 is designed to interface to a DMA controller such as the 8257. The output of the 8275 goes to a character generator ROM such as the 2513 or equivalent. Dot timing and interface circuitry must be provided external to the 8275. A block diagram of an 8275 is shown in Figure 3.

General Operating Characteristics

When using the 8275 in a microcomputer system, a specific section of memory would be set aside for the CRT screen. The 8275 provides what can be thought of as a *window* into this portion of memory. It has two row buffers. While one row is being displayed, the second buffer is being filled with the next row of characters. The row buffers are filled using Direct Memory Access (DMA). The DMA burst length and spacing are programmed during initialization. Special codes are included to reduce DMA overhead.

There are four special codes available:

1. End of Row — The end of row code activates Video Suppression (VSP) and holds it to the end of the line.
2. End of Row-Stop DMA — This code causes the DMA control logic to stop DMA for the rest of the row when it is written into the row buffer.
3. End of Screen — This command will activate VSP and holds it until the end of the current frame.
4. End of Screen-Stop DMA — This code causes the 8275 DMA control logic to end the DMA for the remainder of the frame. The display is affected the same way as End of Screen.

An important note in programming the 8275 is that if Stop DMA is not the last character in a particular burst, the DMA will continue until the next character is read.

Before the start of a frame, the 8275 requests DMA and fills its first row buffer. As the first horizontal sweep begins, characters are output from the row buffer just filled to the character generator. At the same time DMA begins filling the second row buffer with the next row of characters. After buffer number one is scanned once for each line in the row, the roles of the two buffers are reversed and the process is repeated. This procedure is repeated until all character rows are displayed.

The number of rows on the screen and the number of characters per row are programmable during initialization. This allows easy interface to a large number of display types.

Since the 8275 displays each row of characters one line at a time, it must provide data to the character generator as to what line in the character is being displayed. This is done via the line counter outputs $LC_{0\text{-}3}$. The 8275 character generator interface is shown in Figure 4. The number of lines in a row is programmable from one to sixteen. This is sufficient for both upper and lower case letters and underlining.

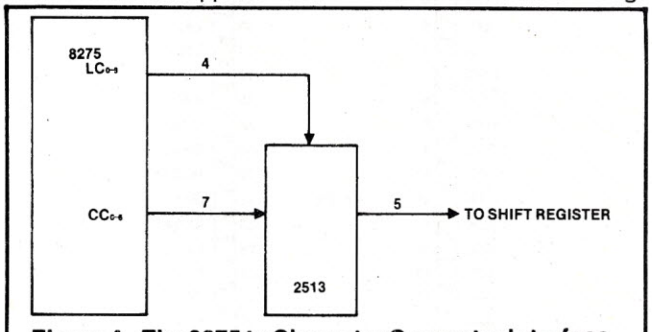


Figure 4. The 8275 to Character Generator Interface.

Special Visual Attribute Codes are provided to cause special action or symbols on the screen without the use of a special character generator ROM. These will be discussed later.

The 8275 has outputs for horizontal and vertical retrace. These signals can be combined with the final video information to interface to a CRT monitor or home television receiver. The timing of these signals is programmable during initialization.

SCREEN FORMAT

The 8275 can be programmed to display from one to eighty characters per row and up to sixty-four rows per frame. The maximum number of characters possible per row for a specific CRT monitor can be calculated from the frequency response of the vertical amplifiers. For example, suppose you desire to have a full eighty character line. The dot counter is designed to divide seven (five character dots plus two spaces). Then there are 560 dot positions per line. By using our original television receiver specification of 53.5 μsec of usable display time per line, we can calculate the minimum required frequency response for our CRT vertical amplifiers.

$$f3db = (560) \quad \frac{1}{53.5 \times 10^{-6}} \quad = 10.46 \text{ MHz}$$

We can generate a formula for the calculation of the number of characters possible on a line for a specific CRT monitor.

$$\frac{\text{(usable display time (}\mu\text{sec)) (f3db)MHz))}}{\text{(\# dots/character)}} = \text{\# of characters/line max}$$

Vertically we have 242 usable lines per frame. If one wanted to evenly divide this number, it would give us a maximum of 22 eleven line rows.

The line count is held stable while outputting the appropriate characters during each horizontal sweep and incremented during horizontal retrace. This procedure is repeated until the entire character row is displayed.

The output of the line counter can be programmed to be in one of two modes. In mode 0 the output of the line counter is the same as the line number. In mode 1 the line counter is offset by one from the line number. Mode 0 is used for character generators whose first line is at row address zero. Placement of the underline is also programmed in during system initialization.

RASTER TIMING

To function properly, the CRT controller must provide signals informing the CRT monitor when to start horizontal and vertical sync. As with all other possibly varying functions, the sync durations are programmable by the host processor.

After the character counter has completed a full line of characters, it starts the horizontal retrace interval. This interval can be programmed to be from two to thirty-two character clocks in duration. This process is repeated until the entire frame has been displayed. The row counter counts the number of complete rows on the screen. The vertical retrace interval is programmable for up to four complete rows (up to 48 lines).

Both horizontal and vertical retrace signals are output from the 8275. These signals must be synchronized with the video signal to the CRT display using the dot timing circuitry.

DMA TIMING

The 8275 can be programmed to request burst transfers of one to eight characters. The interval between bursts is also programmable (from 0 to 55 ± 1 character clock periods). This allows the designer to tailor his DMA overhead to fit his particular system environment. If, for any reason, there is a DMA underrun, a flag is set in the status word. The DMA is usually initialized for the next frame at the end of the current frame.

INTERRUPT TIMING

The 8275 can be programmed to generate an interrupt request at the end of each frame. This can be used to reinitialize the DMA controller. If the 8275 interrupt enable flag is set, it will issue an interrupt request at the beginning of the last display row. Interrupt request will go inactive after the status register is read or a reset command is issued; however, it is not recommended to issue a reset command to the 8275 during normal service.

An alternate method of reinitializing the DMA controller is to have the DMA controller itself interrupt on terminal count. When using this method, be sure to disable the 8275 interrupt enable flag.

Upon power up, the 8275 Interrupt Enable Flag may be set. The user's power up routine should always write a reset command to the 8275 before system interrupts are enabled.

OUTPUT CIRCUITRY

The question now arises, "What else is needed in addition to the 8275 to produce a working CRT display?" For the most part, the interface to the computer side has already been defined. A DMA controller such as the Intel 8275 is required to fetch the data. Additionally, a section of memory equal to the total number of characters to be displayed plus any attributes and special codes is needed.

It is important to understand, at this point, that a specific location on the screen of an 8275 controlled CRT display does not have an absolute address in memory as our original MSI controller had. This is because visual attributes (blinking, reverse video, etc.) are stored within the actual display character field.

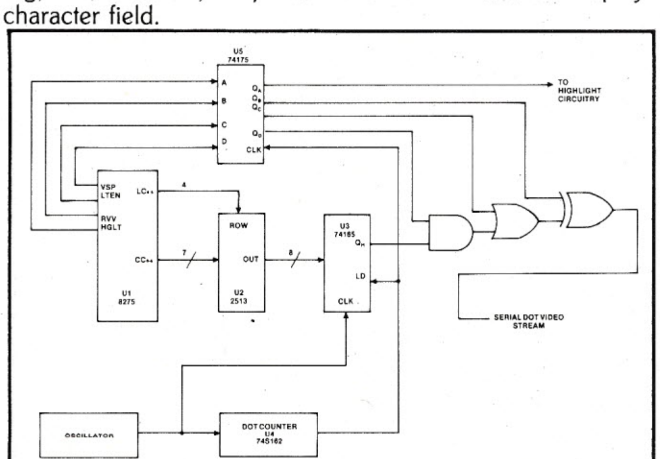


Figure 5. 8275 Dot and Timing Circuitry.

For example, if you wanted half of a particular row to blink and half not to blink, the attribute character which tells the 8275 to start a blinking field is placed after the last character in the non blinking field. This will make the memory field for that row one character longer, offsetting the addresses in the next row.

This will require a certain amount of extra software to handle screen updates but far less than if the CPU had to generate the blinking effect itself. Special hardware can be designed to get around the problem and will be described later.

On the CRT side of the 8275 we do not get away quite as simply. Figure 5 is a basic diagram of the 8275 output circuitry. We will, of course, need a character generator (u2) and an output shift register (u3) to convert the ASCII character information from the processor to the serial dots to be displayed on the CRT screen. A dot counter (u4) is also needed to count the exact number of dots to be shifted out onto the screen before loading. The output of this dot counter, once again, has three functions: it increments the character counter position, it reloads the shift register and shifts out the characters. Intercharacter spacing is a function of the dot counter and length of the shift register.

Some special circuitry will be required to make use of the many features of the 8275 such as blanking, reverse video, blinking characters, highlighting, underlining and limited graphics capability. To make use of the blanking pulses supplied by the 8275 (video suppression pulses), we must provide circuitry that will synchronize this level to the loading of the shift register. This could be a simple D-type register, as shown, clocked by the shift register load pulse.

Reverse video is easy to accomplish on any CRT display. All that is necessary is to place an exclusive-or gate in line with the video information. If the reverse video control line (RVV) is a zero, information will pass through the gate unchanged. If it is a one, the information will be inverted.

Blinking characters are automatically taken care of by the 8275. The video suppression line is used to blink the character or field at a frequency of $\frac{1}{32}$ of the refresh rate.

Underlining is accomplished by the light enable signal (LTEN). Its position in the character block, as has already been explained, is programmed during system initialization. This signal is synchronized and or'ed with the actual video signal (u5 and u6).

Intel has provided another useful feature for field distinction. This is the Highlighting output. Its purpose, as designed, is to make certain characters or fields brighter than the rest of the screen. This is done in the video combiner or modulator circuit. If one does not wish to use it in this manner, it is available for other purposes such as color control if a color monitor is being used.

Visual Attributes, Special Codes and the 8275's Limited Graphics Capability

The characters processed by the 8275 are eight bit quantities. Since the character code outputs from the IC are only seven bits wide, the eighth bit (MSB) is used to determine if it is a normal display character (MSB = 0) or if it is a visual attribute or Special Code (MSB = 1). There are two types of Visual Attribute Codes. They are Character Attributes and Field Attributes.

CHARACTER ATTRIBUTE CODES

Character Attribute Codes are codes that can be used to generate graphics symbols without the use of a character generator. This is accomplished by selectively activating the Line Attribute output ($LA_{0\text{-}1}$), the Video Suppression output (VSP), and the Light Enable output (LTEN). The dot timing and interface circuitry uses these signals to generate the desired symbols. Character attributes, like normal characters, can be programmed to blink or be highlighted. The form a character attribute byte must take is given below.

Table 1.

Character attributes were designed to produce the following graphics:

CHARACTER ATTRIBUTE CODE "CCCC"		OUTPUTS LA$_1$	LA$_0$	VSP	LTEN	SYMBOL	DESCRIPTION
0000	Above Underline	0	0	1	0	┌	Top Left Corner
	Underline	1	0	0	0		
	Below Underline	0	1	0	0		
0001	Above Underline	0	0	1	0	┐	Top Right Corner
	Underline	1	1	0	0		
	Below Underline	0	1	0	0		
0010	Above Underline	0	1	0	0	└	Bottom Left Corner
	Underline	1	0	0	0		
	Below Underline	0	0	1	0		
0011	Above Underline	0	1	0	0	┘	Bottom Right Corner
	Underline	1	1	0	0		
	Below Underline	0	0	1	0		
0100	Above Underline	0	0	1	0	┬	Top Intersect
	Underline	0	0	0	1		
	Below Underline	0	1	0	0		
0101	Above Underline	0	1	0	0	┤	Right Intersect
	Underline	1	1	0	0		
	Below Underline	0	1	0	0		
0110	Above Underline	0	1	0	0	├	Left Intersect
	Underline	1	0	0	0		
	Below Underline	0	1	0	0		
0111	Above Underline	0	1	0	0	┴	Bottom Intersect
	Underline	0	0	0	1		
	Below Underline	0	0	1	0		
1000	Above Underline	0	0	1	0	─	Horizontal Line
	Underline	0	0	0	1		
	Below Underline	0	0	1	0		
1001	Above Underline	0	1	0	0	│	Vertical Line
	Underline	0	1	0	0		
	Below Underline	0	1	0	0		
1010	Above Underline	0	1	0	0	┼	Crossed Lines
	Underline	0	0	0	1		
	Below Underline	0	1	0	0		
1011	Above Underline	0	0	0	0		Not Recommended *
	Underline	0	0	0	0		
	Below Underline	0	0	0	0		
1100	Above Underline	0	0	1	0		Special Codes
	Underline	0	0	1	0		
	Below Underline	0	0	1	0		
1101	Above Underline						Illegal
	Underline		Undefined				
	Below Underline						
1110	Above Underline						Illegal
	Underline		Undefined				
	Below Underline						
1111	Above Underline						Illegal
	Underline		Undefined				
	Below Underline						

*Character Attribute Code 1011 is not recommended for normal operation. Since none of the attribute outputs are active, the character Generator will not be disabled, and an indeterminate character will be generated.

Character Attribute Codes 1101, 1110, and 1111 are illegal.

Blinking is active when B = 1.

Highlight is active when H = 1.

Character Attributes:

MSB LSB
1 1 CCC B H

- H — HIGHLIGHT
- B — BLINK
- CCC — CHARACTER ATTRIBUTE CODE

A listing of Character Attribute Codes is given in Table 1. The additional circuitry required to produce graphics from the Character Attribute signals is shown in Figure 6. u3, u4, and u5 are used to select either the character generator outputs or the decoded graphics information. u6, u7, and u8 are used to actually decode the Character Attribute outputs.

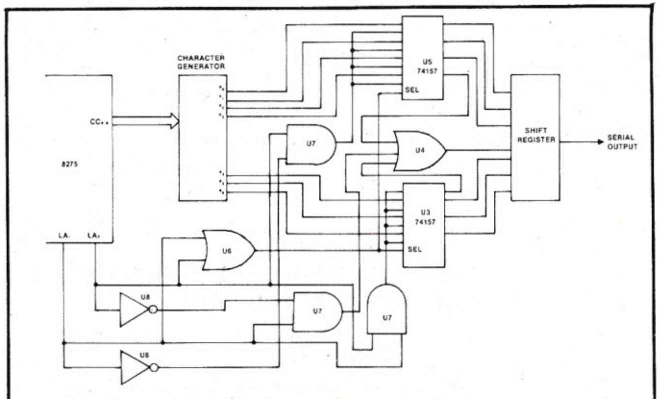


Figure 6. Typical Character Attribute Logic.

FIELD ATTRIBUTES

The Field Attributes are control codes utilized by the 8275 to provide special visual characteristics to the display. They become effective at the character immediately following the control character and remain active up to and including the character which precedes the next field attribute code, or up to the end of the frame. All field attributes are reset during vertical sync. There are six field attributes:

1. Blink — characters will blink at a frequency equal to the refresh rate divided by 32.
2. Highlight — characters will be highlighted by utilizing the highlight output.
3. Reverse Video — character and background colors are reversed.
4. Underline — characters are underlined.

5, 6. General Purpose — These two 8275 outputs act as general purpose, independently programmable field attributes. They are active high outputs and can be used for whatever purpose the designer wishes.

The software configuration of a field attribute byte is given below.

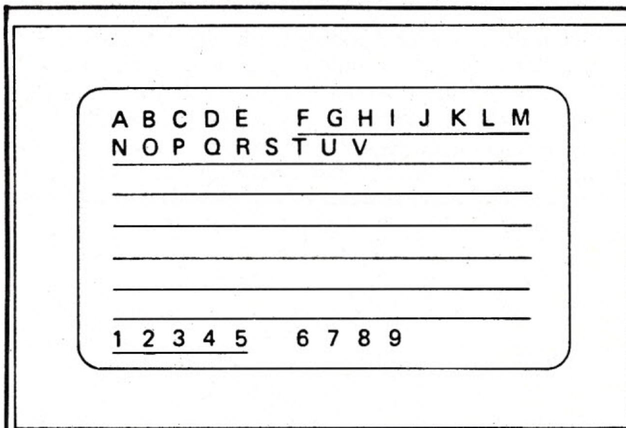
Field Attribute Code:

MSB
1 O U R GG B H

- H — HIGHLIGHT
- B — BLINK
- GG — GENERAL PURPOSE
- R — REVERSE VIDEO
- U — UNDERLINE

All outputs are active high.

The 8275 can be programmed to provide either visible field attributes or invisible field attributes. If the chip is programmed in the visible field attribute mode, all field attributes will occupy a position on the screen. The VSP line will be activated and a blank will appear at the position. Whatever field attribute was selected will appear immediately following the blank. An example of visible field attributes is given in Figure 7.

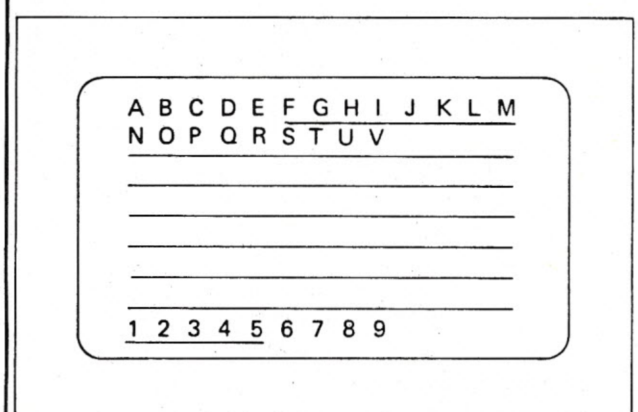


EXAMPLE OF THE VISIBLE FIELD ATTRIBUTE MODE (UNDERLINE ATTRIBUTE)

Figure 7.

When the 8275 is programmed in the invisible field attribute mode, the 8275 FIFO is activated. Each row buffer has its own FIFO. These FIFO's are 16 characters long, limiting the total number of Field Attribute *changes* in any one line to 16. This is what is considered by many engineers in industry to be the major drawback to the 8275.

During DMA, when the row buffer is loaded with a field attribute character, the 8275 recognizes it and places the *next* character in the FIFO. During the display time, when a field attribute is put in the buffer output controller, the 8275 recognizes it again and puts the character in the FIFO on the character code outputs. The chosen attributes are also activated. A special word of warning is necessary here. Since the FIFO is only 7 bits wide, the MSB of any characters put into it are stripped off. Therefore, a Special Code or Character Attribute must not follow a field attribute code. If it does, it will be displayed as a normal character. An example of invisible field attributes is given in Figure 8.



EXAMPLE OF THE INVISIBLE FIELD ATTRIBUTE MODE (UNDERLINE ATTRIBUTE)

Figure 8.

CURSOR CHARACTERISTICS

The cursor position is determined by the current row and position registers which are loaded by the host processor. The cursor can operate in four modes:

1. a blinking underline
2. a blinking reverse video block
3. a non blinking underline
4. a non blinking reverse video block.

The cursor blink frequency is equal to the screen refresh frequency divided by 16.

LIGHT PEN DETECTION

As if all of the features discussed so far were not enough, the engineers at Intel have included a light pen detection circuit. A light pen consists of a micro switch and a tiny light sensor. Circuits for a typical light sensor have been developed and will not be repeated here. When the light pen is pressed against the CRT screen the micro switch enables the light sensor. When the raster sweep crosses the light sensor, it signals the CRT controller.

The 8275 stores the row and character position coordinates in a pair of registers which can be read by the host processor. The 8275 has a bit in its status register which corresponds to the detection of a light pen input signal. The LPEN input must be a 0 to 1 transition for proper operation. Due to the internal delays of the circuitry, the character position coordinates will be off by at least three positions. This must be corrected in software.

PROGRAMMING THE 8275

The 8275 has three internal registers which are of concern to the designer. These are the command (CREG), parameter (PREG), and the status (SREG) registers. The command and parameter registers are used to set up the operational characteristics of the 8275 while the status register contains the internal status of the device.

Two points to remember are that the command register can only be written into and the status register can only be read from. The addressing structure for these three registers is given below:

A0	OPERATION	REGISTER
0	READ	PREG
0	WRITE	PREG
1	READ	SREG
1	WRITE	CREG

The sequence used to program the 8275 is to first give it a command followed by 0 to 4 parameters, depending on what command was just issued. If you do not give it the proper number of parameters before issuing another command, a special status flag is set indicating an improper command sequence.

8275 INSTRUCTION SET

The 8275 will respond to eight commands. They are:

1. RESET
2. START DISPLAY
3. STOP DISPLAY
4. READ LIGHT PEN
5. LOAD CURSOR
6. ENABLE INTERRUPT
7. DISABLE INTERRUPT
8. PRESET COUNTERS

The Reset Command

After receiving a reset command, the 8275 will stop any DMA requests. The interrupt is disabled and video output is

blanked, using the VSP output from the IC. The HRTC and VRTC (Horizontal and Vertical retrace) outputs continue to run, however, until the host processor initializes the 8275, HRTC and VRTC timing is random after power up.

The format of the Reset Command is shown in Table 2.

Table 2.

	Operation	A0	Description	
Command	Write	1	Reset Command	00000000
Parameter	Write	0	Byte 1	SHHHHHHH
Parameter	Write	0	Byte 2	VVRRRRRR
Parameter	Write	0	Byte 3	UUUULLLL
Parameter	Write	0	Byte 4	MFCCZZZZ

The designations given in the parameter blocks are explained below.

1. Spaced Rows (S)

S	Function
0	Normal Rows
1	Spaced Rows

2. Number of Horizontal Characters/Row (HHHHHHH)

HHHHHHH	# of Characters/Row
0000000	1
0000001	2
•	•
•	•
•	•
1001111	80
1010000	UNDEFINED
1111111	UNDEFINED

3. Vertical Retrace Row Count

VV	# of Rows/VRTC
00	1
01	2
10	3
11	4

4. Vertical Rows/Frame (RRRRRR)

RRRRRR	# of Rows/Frame
000000	1
•	•
•	•
•	•
111111	64

5. Underline Placement (UUUU)

UUUU	Line # of Underline
0000	1
0001	2
•	•
•	•
•	•
1111	16

6. Number of Lines per Character Row (LLLL)

LLLL	# of Lines/Row
0000	1
0001	2
•	•
•	•
•	•
1111	16

7. Line Counter Mode (M)

M	Line Counter Mode
0	Mode 0 (non-offset)
1	Mode 1 (offset by 1 count)

8. Field Attribute Mode (F)

F	Field Attribute Mode
0	Transparent
1	Non-transparent

9. Cursor Format (CC)

CC	Cursor Format
00	Blinking reverse video block
01	Blinking underline
10	Nonblinking reverse video block
11	Nonblinking underline

10. Horizontal Retrace Count

ZZZZ	# of Character Counts/HRTC
0000	2
0001	4
•	•
•	•
•	•
1111	32

The Start Display Command

This command enables the interrupts and video. DMA requests begin and the video and interrupt enable status flags are set.

	Operation	A0	Description	Data MSB	Bus	LSB
Command	Write	1	Start Display	001	SSS	BB

Burst Space Code = SSS

	# of Character Clocks Between DMA Bursts
000	0
001	7
010	15
011	23
100	31
101	39
110	47
111	55

Burst Count Code = BB

	# of DMA Cycles per Burst
00	1
01	2
10	4
11	8

The Stop Display Command

This command disables the video, the interrupts remain enabled, HRTC and VRTC continue to run, and the Video Enable Status Flag is reset, the "Start Display" command must be given to re-enable the display.

	Operation	A0	Description	Data Bus MSB LSB
Command	Write	1	Stop Display	010 000 00

The Read Light Pen Command

This command signals the 8275 to supply the contents of the light pen position registers in the next two read cycles of the parameter. During the first read cycle, the character number within the row is given. During second read cycle, the row number is given. Do not forget that software correction of the character position is necessary due to the internal delays.

	Operation	A0	Description	Data Bus MSB LSB
Command	Write	1	Read Light Pen	01100000
Parameter	Read	0	Character #	(Char. Pos. in Row)
Parameter	Read	0	Row Number	(Row Number)

Load Cursor Position

The load cursor position command conditions the 8275 to put the next two parameter bytes into the cursor position registers.

	Operation	A0	Description	Data Bus MSB LSB
Command	Write	1	Load Cursor	10000000
Parameter	Write	0	Char. Number	(Char. Pos. in Row)
Parameter	Write	0	Row Number	(Row Number)

Enable Interrupt

The Enable Interrupt Command enables the interrupt enable status flag and interrupt.

	Operation	A0	Description	Data Bus MSB LSB
Command	Write	1	Enable Interrupt	10100000

Disable Interrupt Command

The Disable Interrupt Command resets the interrupt enable input status flag and disables the interrupt.

	Operation	A0	Description	Data Bus MSB LSB
Command	Write	1	Disable Interrupt	11000000

Preset Counters Command

The Preset Counters Command sets the internal registers to the display in the left, uppermost corner. Two character clocks are required for this operation. The counter will stay in this state until any other command is given. This command is useful for system debug and synchronization of clustered CRT displays on one CPU.

	Operation	A0	Description	Data Bus MSB LSB
Command	Write	1	Preset Counters	11100000

THE STATUS REGISTER

The status register is a register within the 8275 which contains operational information on the unit. The read command as well as a description of each flag are shown in Table 3.

Table 3.

	Operation	A0	Description	Data Bus MSB LSB
Command	Read	1	Status Word	O IE IR LP IC VE DU FO

IE — (Interrupt Enable) This flag is set or reset on command. It enables the vertical retrace interrupt.

IR — (Interrupt Request) This flag is set at the beginning of the last row of the frame if the interrupt enable is set.

LP — (Light Pen Detection) This flag is set when the light pen input (LPEN) is activated and the light pen registers have been loaded.

IC — (Improper Command) This flag is set when a command parameter string is too long or too short.

VE — (Video Enable) This flag indicates the CRT is enabled for video operation.

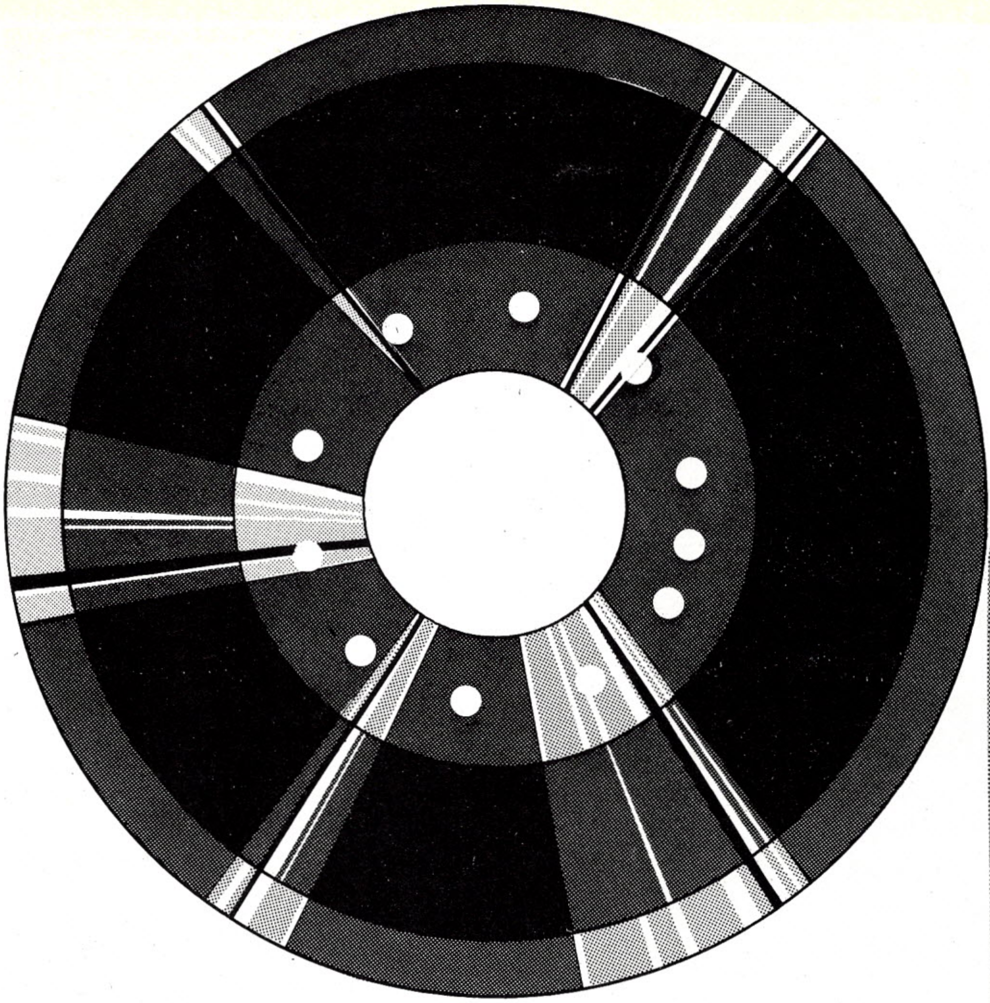
DU — (DMA Underrun) This flag is set whenever data underrun occurs during a DMA operation. When the 8275 gets a DU, the screen is blanked until the next vertical retrace interval.

FO — (FIFO Overrun) This flag is set when the internal FIFO overflows.

THE 8275 MEMORY ARCHITECTURE

The 8275 operates in two modes, the Visible Field Attribute mode and the Hidden Field Attribute mode. The Visible Field Attribute mode is easy enough to handle in software. When a field attribute change is desired (for instance, if you want to start underlining characters), the code is placed in memory at the location of the attribute change. During display the location of the attribute change starts at the next screen position. This gives a nice one to one correspondence between memory locations and screen locations.

The Hidden Field Attribute mode, as previously described, requires the desired field attribute change to be located at the boundary of the fields. However, no blank position is left on the screen. This means that there is one more memory location required per screen attribute change per line. One to one correspondence between screen locations and memory locations is no longer possible, and more elaborate software is required to support the display. This problem can be overcome by complicating the hardware to a certain degree.



It isn't in the hole it's on the doughnut!

INTRODUCING — LOUMAR MANAGEMENT SYSTEMS SOFTWARE

4 programs to give you a comprehensive package for business management

The Loumar General Accounting System is a versatile, fully integrated software package designed for small and medium sized businesses. It is also suitable for CPA's and bookkeeping service films.

The complete software system is composed of four main modules: GENERAL LEDGER, ACCOUNTS RECEIVABLE, ACCOUNTS PAYABLE AND PAYROLL. Each module may be used separately or in combination with any other module. Supplied on disk as run-time modules. Source not available.

All software is written in CBASIC II and utilizes the powerful CP / M operating system.

General system features include:

Automatic posting to general journal • Strict error detection • Report production on demand • Consistent operating procedures • User oriented. No previous computer knowledge required • Designed by accounting professionals • Comprehensive, well presented reports and manuals • Single or multiple client capabilities.

HARDWARE SPECIFICATIONS

The end user's microcomputer must satisfy the following requirements:

48k RAM

Dual floppy disk system

Printer with tractor. All printing is done in 80 col. format

CRT with at least a 64 character by 16 line display

CP / M and CBASIC II

Write for our brochure — Dealerships still available

Contact: Distributor

PAYROLL: Up to 500 employees — **$550.**

ACCOUNTS RECEIVABLE: Up to 1000 customers and 1000 monthly transactions — **$550**

ACCOUNTS PAYABLE: Up to 1000 vendors and 1300 monthly transactions — **$400**

GENERAL LEDGER: Up to 200 accounts with 2000 entries — **$450**

MISSION CONTROL • 2008 WILSHIRE BOULEVARD, SANTA MONICA, CA 90403 • (213) 829-5137

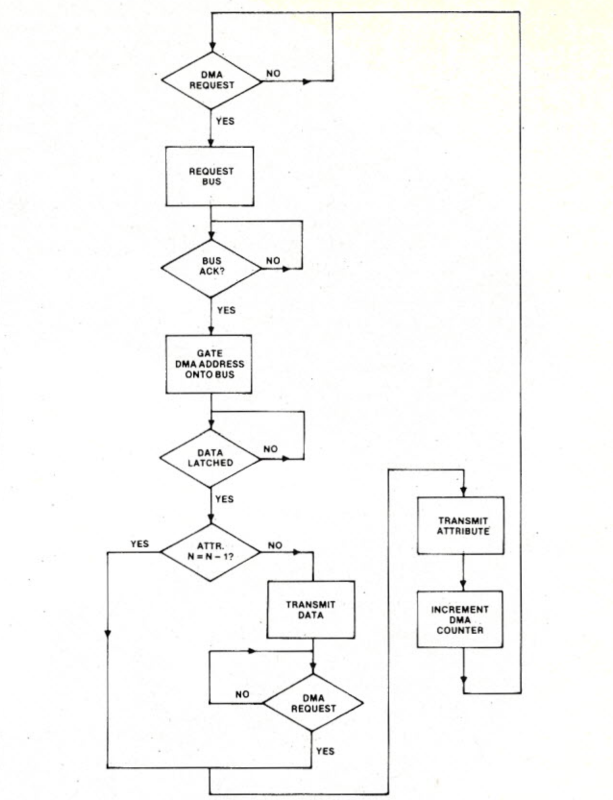


Figure 9. Flow Chart for the Special Purpose DMA Controller.

To allow for correspondence between screen locations and screen data locations in refresh memory in the hidden attribute mode, it is necessary to create a second block of memory for attributes equal in length to the screen data memory. In other words, for each screen data location in memory there is a corresponding screen attribute location. Each time the 8275 issues a DMA request to the DMA controller, special logic checks to see if the attribute at that location is the same as the last location. If it is, the data is written into the 8275 line buffer and the DMA counter is incremented. If it is not, the attribute is sent to the 8275, the DMA counter is not incremented, and the logic sends the data on the next DMA request. A flow chart for this function is presented in Figure 9. The DMA controller function was included in this flow chart since it seems natural to combine the two functions into one.

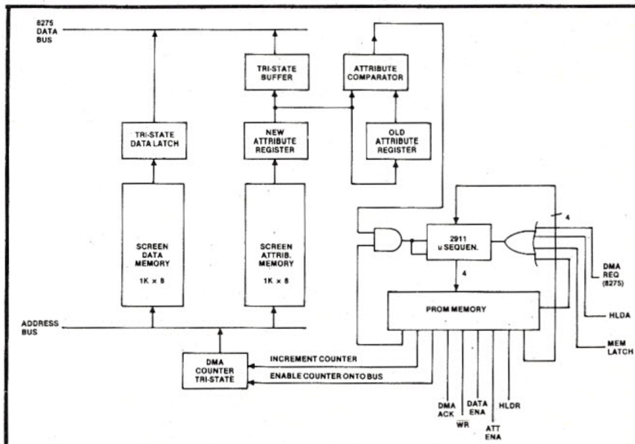


Figure 10. A Special Purpose DMA Controller.

A block diagram of one possible implementation of this function is shown in Figure 10. A 2911 microprogram sequencer is used to control program flow while two 8223 32x8 PROMs hold the actual instructions. When the DMA

controller addresses screen memory, data is latched into an 8-bit register, the new attributes are latched into a second register while the old attributes are transferred from the second register to a third.

A comparison is made between the two sets of attributes, and the result is gated into the S_0 and S_1 inputs of the 2911. According to the 2911 data sheet, if S_0 and S_1 are both logical "0", the microprogram counter is selected. If S_0 and S_1 are both logical "1", the direct address inputs are used (as in a branch instruction). This gives the ability to jump if the attributes are equal.

The Carry In (Cn) input to the 2911 is used to hold the sequencer on an instruction while we are waiting for a DMA request from the 8275 or HLDA from the 8080A. If Cn is low, the microprogram address is not incremented. If it is high, the address is incremented by one.

All other functions of the DMA controller are handled by the outputs of the PROMs. These include gating information onto the bus, incrementing the DMA address counter, and supplying control signals to the 8080A and the 8275.

Since a DMA controller is always required by the 8275, the additional logic added when using this method does not add significantly to the size of the system. It may even reduce the cost when one considers the price of the 8257 DMA controller suggested by Intel.

SCROLLING THE DISPLAY

Scrolling, when using the 8275, is simple. Each time the 8275 finishes a refresh of the screen (sixty times a second), it interrupts the 8080A microprocessor. At this time the 8080A must re-initialize the DMA address counter. All that is required to scroll up one line is to change this initialization by the number of locations in one line. This feature allows scrolling both up and down, allowing inspection of what has already gone off the screen.

SOME SOFTWARE HINTS

To define the location of a character on the screen, two pointers, row count, and column count are created in memory. Row count represents the position of the first character in a given row. For the first row, a row count equals 0000H. For the second row, row count equals 0050H (0080D) for an 80 character line. Column count represents the column in which the character is located. Character position on the screen can be calculated by adding the row count to the column count.

There must be a relationship between the row and column count pointers and the 8275 cursor X and Y position registers. Cursor X will be exactly the same as column count. Cursor Y will be equal to row count divided by the number of characters in a row for your particular system.

There is one last parameter needed to complete the display memory addressing procedure. This parameter is called TOP and is the address in memory for the upper left hand character on the screen. All that is necessary to locate a character in memory is to add its row number, character number within the row, and location of the start of display memory (TOP). The parameter TOP is the value which is modified when scrolling is desired.

For an in depth discussion of setting up a CRT routine refer to "CRT Terminal Design Using the Intel 8275 and 8279" by John Murray and George Alexy (Intel Application Note #AP-32).

SUMMARY

I have attempted to present a detailed explanation of the 8275 with enough system information to allow an experienced experimenter to use one in his system. The IC is easy to use and available now. A great deal of creativity is possible in the actual system design and none of the designs I have shown are necessarily the only way of implementing the desired function.☐

The Pascal Notebook

Chapter 1

By Henry Davis, Associate Editor

PREFACE

For many years compiler writing has been a secret ritual practiced by few programmers. Because few people write compilers, there is often an air of mysticism surrounding a compiler and the compiler writing task. Writing a compiler for a language like Pascal really isn't difficult because the programming techniques are simple.

What sets compiler writing apart from other programming tasks are the size (the one presented in this series is about 5,000 lines of code), the variety of programming techniques used, and the high visibilty that the implementation has with other programmers. These factors require that good programming practices be used throughout the implementation. Additionally, the compiler writer understands not only the mechanics of the language but also the programming philosophy and intent of the language designer.

Because implementation of Pascal requires much more than a simple understanding of the language, there is no better way to learn the full meaning and impact of the language. Throughout this series we'll look at the "whys" of compiler writing and in most cases the "hows" will fall directly out.

The compiler will be presented in a top-down manner so that the implementation details get pushed to last. This is done so you will know where you're headed and can measure your progress.

SERIES OUTLINE

Because Pascal is a language, the concepts of syntax and semantics are presented first. Even though the mathematics may seem difficult at first, a basic understanding of these subjects will aid greatly in understanding Pascal. However, don't be scared off by the formal language material because it is shown in a practical manner later. Next the subject of parsing is considered and a general parsing algorithm is developed. The purely theoretical material is concluded with a section on BNF notation, syntax graphs and using these tools to write language specific parsers.

The actual introduction of Pascal begins with programming equivalents of the syntax graphs and then continues into basic data structures and how they are defined. As a prelude to discussing the actual compiler and the fine points of Pascal, a complete syntax definition is presented in both BNF and syntax graphs. The compiler is presented in pieces beginning at the highest level of abstraction and then filling in the details. Topics presented in order are:

1. Outer compiler structure and initialization
2. The scanner
3. Blocks and declarations
4. Constants, types, id search
5. Parameter list, field list
6. Code generation
7. Type checking, statements
8. Expressions
9. Control structure
10. Intrinsic functions
11. Interpreting the compiled code
12. Native code generation

Each new programming technique is explained at the time of usage so your knowledge of Pascal will grow with your understanding of compiler writing. In particular, the subtleties of Pascal must be explicitly dealt with in the compiler. Not all of the material in this series is easy to master, but once understood it will improve your programming skills. To get the most out of each chapter first read the material all the way through, marking points you don't understand. Then reread the chapter and attempt to answer your questions. Finally proceed to any program examples and follow them in depth. You will find the material easier as the series progresses so don't be discouraged at the beginning if the going gets rough. If a point eludes you, after careful study, my address and phone number are included at the end of each chapter.

Road Map — Chapter 1

INTRODUCTION

Programming languages have traditionally been introduced to the programming community on a "mechanical" basis. That is, the languages have been presented on a "how to use it" level. Part of the reason for this ad hoc approach is that most languages, before Pascal, have been designed without must regard for the compiler writer, or how the programming techniques can be explained to the user. This isn't to say that other programming languages are poor. These languages have filled a need, and usually perform well in their intended application.

Programming languages have evolved from basic assembly language to Pascal in less than thirty years. As good as Pascal is, there is, and will be, room for improvement as we learn more about the "art of programming." To learn more about Pascal, we'll start with the motivating circumstances and include a little theory. The final goal of this series is to detail a Pascal translator, written in Pascal, so that you can implement your own compiler. Additionally, you'll know enough about language design to specify your own.

HISTORICAL BACKGROUND

In 1968 a preliminary version of the programming language Pascal (named after the 17th century mathematician Blaise Pascal) was drafted. Similar to ALGOL-60 and ALGOL-W, Pascal is based to a large extent on the contributions by E.E. Dijkstra and C.A.R. Hoare. Dijkstra's work *Notes on Structured Programming*[1] formally presented a programming methodology that put "science" in the "art." Hoare's work, *Axiomatic Basis of Computer Programming,*[2] showed that the properties of programs could be rigorously proved by using mathematical reasoning. These papers focused on methods of representing algorithms by program text. Additional papers[3,4,5,6] applied these same techniques to data structures. Continued work by Niklaus Wirth and others at ETH, Zurich, Switzerland, resulted in an operational compiler in 1970 and a revised report in 1973.

SYNTAX AND SEMANTICS

The job of a compiler is to translate from one language (source code — in this case Pascal) to another (object code or some intermediate language). Any programming language is simply a concise and unambiguous way to express a method of doing a task. In fact, a programming language is a highly restricted subset of natural language (like English) which usually uses a compact representation (like mathematics) for common notions.

In many ways, programs are like cooking recipes; each uses terminology and form specific to the application. Because programming languages are a restricted form of natural language, it makes sense to use linguistical tools to learn as much as possible about Pascal. Understanding the fundamentals of languages and how they impact both language design and compiler implementation will greatly aid in understanding Pascal and compiler writing in general.

Pascal, like all programming languages, can be described by means of formal languages and analyzed using language theoretic techniques. Each language is based on a set of symbols called a vocabulary.

In ordinary language, like English, the symbols of the language are words. Because these words are the most basic and elemental pieces of the language, formal languages refer to them as "terminal symbols." The number of symbols in a vocabulary (also called an alphabet) is arbitrary but always finite. This means that it makes sense to talk about the *ith* element (word) in the set. A sentence "over an alphabet" is a collection of symbols from the alphabet composed as a string. The string must be of finite length or no length. A sentence with no length (having no symbols) is called the empty sentence. A space is not an empty sentence since it has a length, one. A language, like English, is just a set of sentences.

Mathematicians and computer scientists use a shorthand representation for these and other definitions. Lest the terminology and symbology of the "high priests" of programming languages intimidate you, example eight reduces all the following theory to practice. However, reasonable effort should be expended in attempting to understand the theory, since it is the theory which motivates Pascal.

Languages are referred to by using a capital L, possibly with subscripts when talking about different languages. An alphabet or vocabulary is usually written as a capital V. V is a set of symbols so it makes sense to talk about some properties based upon operations on the set. V^* is used to represent the set of all sentences formed using symbols of V. The empty sentence, E, is included in V^*.

As an example of this notation:

let the vocabulary have symbols:
THE,BIG,DOG,SAT,QUIETLY
or V = THE,BIG,DOG,SAT,QUIETLY
The possible sentences are: THE,BIG,DOG,SAT, QUIETLY, THE DOG SAT, . . .
or V^* = THE,BIG,DOG,SAT,QUIETLY,THE DOG SAT, . . .
When the language is a subset of V^* say: THE DOG SAT
or L = THE DOG SAT
Another language L_1 = THE BIG DOG SAT, THE DOG SAT QUIETLY

Sometimes it is convenient to exclude the empty sentence, E, from consideration, so we write $V^+ = V^* - E$, meaning the set includes all sentences except the null string (empty sentence).

To summarize in mathematical definitions (an example follows):

D.1 Let a language L=L(V,N,P,S) be defined by
- (a) a vocabulary V of terminal symbols
- (b) a set of N non-terminal symbols ("grammatical categories")
- (c) a set P of productions (syntactic rules)
- (d) a start symbol S which is from N

D.2 The language L is a set of sequences of terminal symbols ξ (verbalized as Xi) that can be generated (produced) from S according to D.3. Mathematically:

$$L = \{\xi \mid S \xrightarrow{*} \xi \text{ and } \xi \in V^*\}$$

verbalized as "the language L is a set of Xi such that S generates Xi and Xi is an element of the complete set of sentence."

D.3 A sequence σ_n (verbalized as sigma sub n) is generated from a sequence σ_0 if and only if there exist sequences $\sigma_1,\sigma_2,\ldots,\sigma_{n-1}$ such that every σ_i is directly generated from σ_{i-1} according to D.4. Mathematically, verbalized as "sigma sub zero generates sigma sub n if and only if sigma sub (i-1) directly generates sigma sub i for all i from 1 to n."

D.4 A sequence σ is directly generated from a sequence if and only if there exist sequences such that:
- (a) $\sigma_i = \alpha\, \sigma_i'\, \beta$
- (b) $\sigma = \alpha\, \sigma'\, \beta$
- (c) and P includes the production $\sigma_i \rightarrow \sigma'$

To illustrate what these definitions mean, let's consider an example from ordinary English. English teachers used to include sentence diagrams as a fundamental part of an English course. The sentence "The big dog sat quietly" can be diagrammed as in Figure 1. To produce the diagram, the sentence is parsed (analyzed, or traced according to the generating steps) by first noting that it is a "sentence" (indicated by <sentence>).

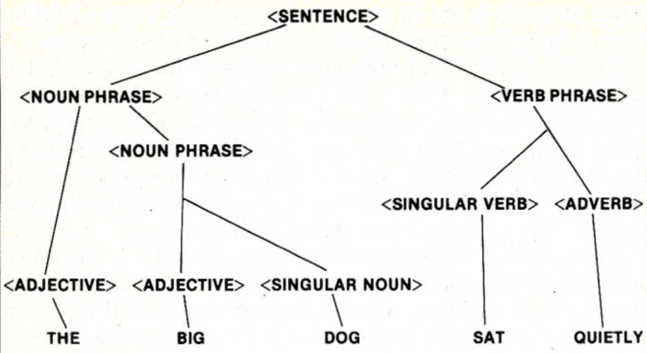


Figure 1.

The sentence is further broken down into a noun phrase, "The big dog," and a verb phrase, "sat quietly." Now break the noun phrase into its basic parts; the singular noun, "dog," modified by the adjectives, "The," and "big." Further parse the verb phrase into the singular verb, "sat," and the advert, "quietly." The result yields exactly the diagram of Figure 1.

The production rules, "P", that we used in parsing this sentence can be written as:

E1.1 <sentence> → <noun phrase> <verb phrase>
E1.2 <noun phrase> → <adjective> <noun phrase>
E1.3 <noun phrase> → <adjective> <singular noun>
E1.4 <verb phrase> → <singular verb> <adverb>
E1.5 <adjective> → The
E1.6 <adjective> → big
E1.7 <singular noun> → dog
E1.8 <singular verb> → sat
E1.9 <adverb> → quietly

The meaning of these lines is:

1. A "sentence" is formed from a "main phrase" and a "verb phrase."
2. A "noun phrase" is formed from an "adjective" and a "noun phrase."
3. A "noun phrase" is formed from an "adjective" and a "singular noun."
4. A "verb phrase" is formed from a "singular verb" and an "adverb."
5. An "adjective" is the word "The."
6. A "adjective" is the word "big."
7. A "singular noun" is the word "dog."
8. A "singular verb" is the word "sat."
9. An "adverb" is the word "quietly."

The arrow in these rules means that the item on the left can produce or generate the item on the right. The names of the parts of the sentences are enclosed in brackets to avoid confusion with the English words. These syntactic categories (<sentence> etc.) are the nonterminals, N, of this language. The actual words, "The," etc., are the terminals, V. The relations defined by the rules are the productions, P, including a set of productions that generates exactly the terminals of the language. <SENTENCE> is the start symbol.

Within the framework of the definitions D.1:

L=L(V,N,P,S)
V={THE,BIG,DOG,SAT,QUIETLY}
N={<SENTENCE>, <NOUN PHRASE>VERB PHRASE>, <ADJECTIVE>, <SINGULAR NOUN>, <SINGULAR VERB>, <ADVERB>}
P=the set of rules E1.1-E1.9
S=<SENTENCE>

D.2 L is an infinite set, one element of which is the sentence we have considered. Note that "BIG BIG BIG THE DOG SAT QUIETLY" is also a grammatically correct sentence in the broadest sense of the meaning. The key here is that we have dealt only with syntax (structure) but not with meaning (semantics).

D.3 <SENTENCE> generates "THE BIG DOG SAT QUIETLY", since there is a sequence of productions which terminates in this sentence.

D.4 <ADVERB> → QUIETLY and <NOUN PHRASE> → THE BIG DOG are examples of direct generation.

Note that a language and its associated grammar can be used to both recognize (parse) and generate sentences.

The fundamental task of language translators in general, and compilers in specific, is to recognize, not to generate, sentences. To recognize a sentence it is necessary to reconstruct the productions which resulted in that sentence. The complexity of this task directly relates to the type of productions which define the language.

Historically, compiler writers have been handed a language definition and forced to implement exactly that language. Pascal was based on an easy algorithm of syntax analysis. At this point let's follow Wirth's work and define an efficient algorithm for parsing, and then see what class of languages can be handled.

In order to meet the basic requirement of efficiency, the production rules are restricted to those which depend only on the present state of the parsing algorithm and the value of the next symbol. This means that it is not necessary to "stack up" information waiting to make a decision.

Along the same line of reasoning, once at a step it must not be necessary to revoke that or any other previous decision. These two requirements are known collectively as one-symbol-lookahead without backtracking. An example of backtracking is found in FORTRAN:

DO 10 J=1,10

Until the comma between 1 and 10 is found, it is impossible to determine if DO 10 J is a variable or the beginning of a DO statement. Many compilers assume one or the other and go back if the wrong decision was made.

The basic methodology we will pursue is called Top-Down Parsing. This technique consists of trying to reconstruct the production from its start symbol to the final sentence. The sentence is successively partitioned from the top down, using the most general rule appropriate at each step, until no productions are left.

To develop a parsing algorithm, let's start with Example 1 and see how the sentence is formally parsed. The problem we must solve is whether or not the sentence belongs in the language.

Definition 2 requires that any sentence be produced from the start symbol. In this case it must be generated from the start symbol <SENTENCE>. Rule E1.1 states that a sentence is only valid if it is a noun phrase followed by a verb

<sentence>	The big dog sat quietly
<noun phrase< <vb. phrase>	The big dog sat quietly
<adj.> <noun phrase> <vb. phrase>	The big dog sat quietly
The <noun phrase> <vb. phrase>	big dog sat quietly
The <adj.> <sing. noun> <vb. phrase>	big dog sat quietly
The big <sing. noun> <vb. phrase>	dog sat quietly
The big dog <verb phrase>	sat quietly
The big dog <sing. verb> <adverb>	sat quietly
The big dog sat <adverb>	quietly
The big dog sat quietly	

Figure 2.

phrase. At this point two sub-tasks have been defined: analysis of the noun phrase and verb phrase. First determine if any portion of the sentence can be generated from the symbol <NOUN PHRASE>. By applying rules E1.2 and E1.4 "THE" can be generated.

By continuing in this fashion and marking each generated symbol off the input sentence, the sentence is accepted. Acceptance occurs when the input sentence is completely processed. Figure 2 illustrates the steps in the parsing procedure. This example actually illustrates the backtrack principle. In particular, there is no way to decide between E1.2 and E1.3 by simply looking at the next symbol. The decision is made by looking two symbols ahead.

As a procedure, this is effected by following one of the possible paths and backing up to the previous decision point if no further progress is possible. In general, there is no upper limit on the number of backtracks which may have to be made. Because of the inefficiencies associated with backtracking, language structures that lead to backtrack requirements must be identified and avoided in good programming languages. The restriction barring backtracking is manifest in the condition that alternate right parts of productions have different initial symbols. This restriction can be stated as follows:

Rule R1

Given the production

$A \rightarrow \zeta_1 \mid \zeta_2 \mid \dots \zeta_n$

(this is shorthand for writing each ζ_i as $A \rightarrow \zeta_i$ for $i = 1$ to n, the bar is verbalized as "or")

the sets of all initial symbols of all sentences produced from each ζ_i must be disjoint.

To show the full impact of the backtracking problem, let's consider an example.

Example E2

E2.1 S→Z|Y
E2.2 Z→XZ|P
E2.3 Y→XY|R

In parsing the sentence XXXR it is impossible to decide whether to apply E2.2 or E2.3 by looking at the first symbol. Rule R1 has been violated since X is the first symbol in the right side of both E2.2 and E2.3. By finding an equivalent set of productions, it is possible to parse Example 2. One such set is:

Example E3

E3.1 S→A|XS
E3.2 A→P|R

Now the sentence XXXR can be parsed in the following manner:

Step 1. We know that S→XS (E3.1), and that we may apply this rule repeatedly if necessary. Applying it once gives us S→XS, twice S→XXS, and three times S→XXXS. In these applications we are simply substituting the definition of S for the X in XS.

Step 2. We now have S→XXXS. If we apply the other half of E3.1, S→A, we produce S→XXXA.

Step 3. Applying E3.2, A→R, now produces S→XXXR.

While Rule R1 eliminates the backtracking issue, another difficulty known as the null string problem crops up. In this situation a dead end is reached in the parsing procedure due to the way in which a non-terminal produces E, the null string. Example E4 illustrates the effects of the null string problem.

Example E4

E4.1 S→BX
E4.2 B→X|E

where E represents the null sequence of symbols, and X is a sentence in this language.

In parsing the sentence X, a dead end can be reached as follows:

Step 1. Starting with S→BX and applying E4.1 produces S→XX

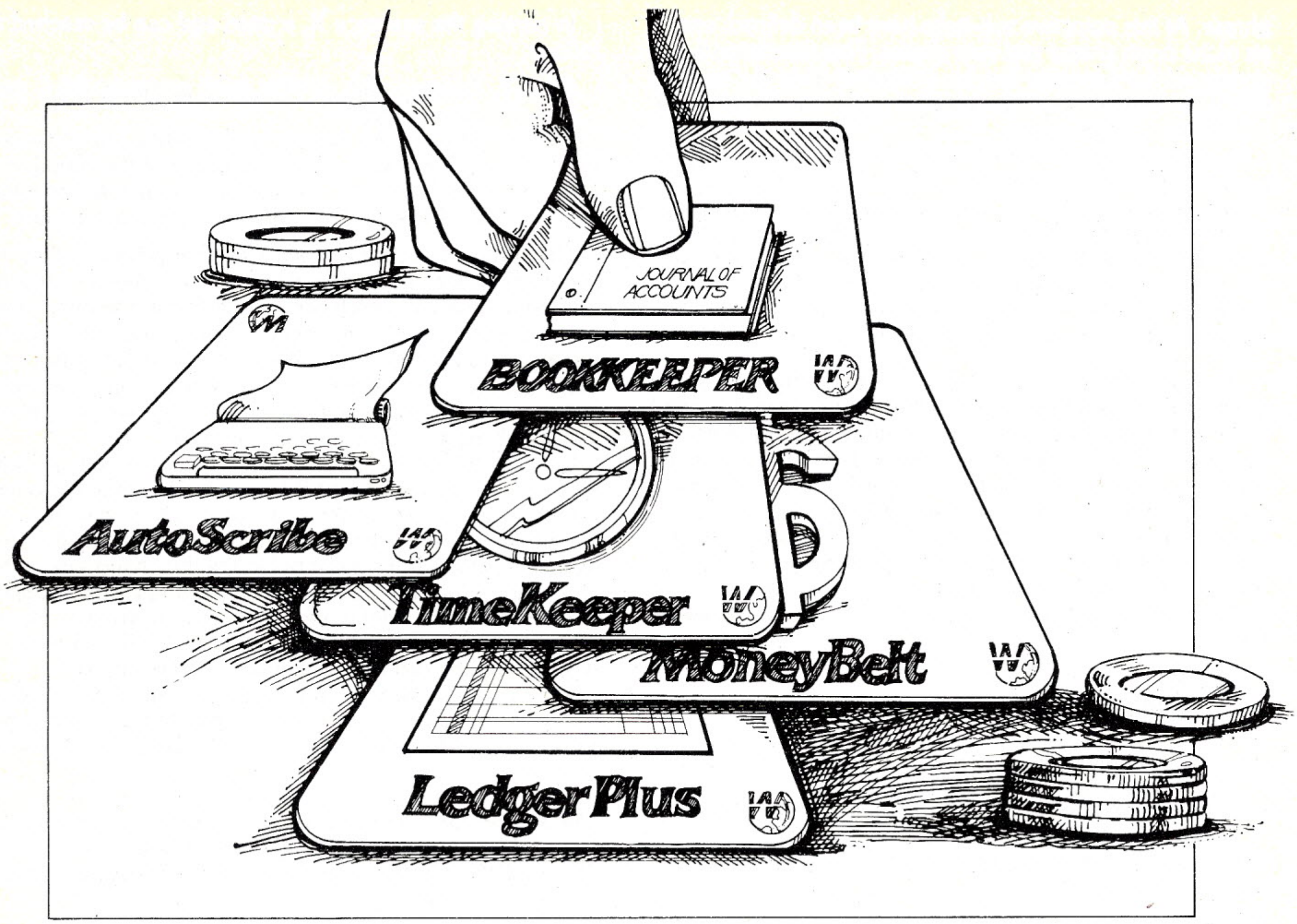
Step 2. There is no way to further parse this statement without backtracking, because of the production of an extra X. This leaves a generated terminal symbol without a corresponding input symbol.

The problem occurred because the production B→E should have been used instead of B→X. In order to avoid the null string problem, which occurs only for non-terminal symbols that can generate the empty sentence, we add a second restriction to the allowable language. Rule R2 states that every non-terminal which generates the empty sentence must have initial symbols different from the set of symbols that are allowed to follow the non-terminal.

Example E4 violates Rule R2 because X can both precede and follow B. Rules R1 and R2 combine to prohibit productions that are known as left-recursive. Such productions are termed left recursive because the non-terminal used in the recursion is the left-most symbol in the right hand side of the production. The real problem stems from not knowing what symbols are to the right of the recursive non-terminal. By rewriting E4.1 as S→X|A the problem of left recursion is avoided and the definition is known as right recursive.

Another way to define production is to extend the symbolism of the meta-language to include facilities to define replication of a symbol. As we will see later, BNF includes such an extension.□

The author can be contacted at American Microsystems, Inc., 3800 Homestead Road, Santa Clara, CA 95051. Phone (408) 246-0330, extension 874. All references will be published in the final chapter.



IN THE HIGH STAKES GAME OF BUSINESS SOFTWARE,

TRUST YOUR DEALER

MicroSource™ is becoming the chosen business software producer. For all the right reasons. For example, before you ever see one of our applications, it's field tested and refined. Another reason is our Operations Manuals. They're called the finest in the industry . . . you never need special training to use MicroSource. One more reason: versatility. MicroSource works with different terminals and printers. You can use your favorite terminal, and choose between speed or print quality in a printer. Add in the rest of the story — like support, enhancements, low cost — and you get the picture.

If your dealer stakes his business on MicroSource software, can you afford to do less?

MoneyBelt — The Money Manager™
A modular in-house bookkeeping package with general ledger, accounts payable and receivable, and payroll. Buy at your own pace . . . one at a time or all at once. Less than $8000 (under $200/mo. on a financing lease) for all hardware and first module (sugg. list). Horizon and other North Star disk based systems.

LedgerPlus — The Company Bookkeeper™
Also modular, with all the accounting modules of MoneyBelt, plus inventory, check reconciliation and mailing list. Each interacts with the others, but stands alone, too. A first for Vector Graphic, MZ, Apple and Micropolis disk based systems. Less than $8000 (under $200/mo. on a financing lease) for all hardware and first module (sugg. list).

Other business software packages by MicroSource: **AutoScribe — The Paperwork Manager™**, **Bookkeeper — The Office Accountant™**, **TimeKeeper — The Time Accountant™**.

MICROSOURCE

1425 W. 12th Place • Tempe, Arizona 85281 • (602) 894-9247 • Cable: MICROAGE
Telex: 165 033 (MICRO AGE TMPE)

The following are trademarks of The Phoenix Group, Inc.:
MicroSource, Autoscribe, Bookkeeper, MoneyBelt, LedgerPlus, TimeKeeper.

DEALERSHIPS STILL AVAILABLE

CIRCLE INQUIRY NO. 80

National Technical Schools Mini-Series of Basic Electronics Unit 4

By Walter F. Stephens
Assistant Chief Instructor
National Technical Schools, Los Angeles, California

TABLE OF CONTENTS

Part 1 of Unit Three presented us with an overview of operational essentials such as number systems, binary arithmetic and logic elements. Unit Three concluded with a discussion of Boolean algebra. This month, simplification of logic circuits will be explained along with the discussion of flip-flops, basic circuitry and introduction to logic families.

SWITCHING SIMPLIFICATION

The complexity of a gating network can be reduced by use of four general reduction techniques:

- Construction of a truth table and analysis of the table to see if there are any simple corrections that can be minimized.
- Developing of a Boolean expression based on a truth table.
- Reduction of the Boolean expression to its simplest form.
- Graphic representation of the logical expressions to make simplification apparent.

In the past, when logic circuits were composed of discrete (separate and distinct) components, great effort was put forth to minimize the number of components and gates in a logic circuit. In today's world of integrated circuits, there are techniques combined with hardware (ICs) that are employed by the logic designer to quickly and simply produce the desired function. Among the choices of integrated circuits utilized to accomplish this are the "Data Selector" and READ ONLY MEMORY (ROM) Arrays. Since the use of these newer techniques require a more advanced IC background than we have accrued here, an old standby, the Karnaugh method, shall be presented.

MINIMIZATION BY THE USE OF KARNAUGH MAPS

A Karnaugh map is a graphical device designed to aid in circuit simplification. It can be thought of as a convenient means for the visual representation of a two-state function, and is used to obtain a simplified Boolean expression from a truth table. The map method is usually employed to reduce a function of three or four variables. If the function has more variables, this method becomes unwieldy.

TRUTH TABLES AND BOOLEAN FUNCTIONS

Implementation of a circuit corresponding to a given Boolean function becomes a straightforward procedure. As a general rule, however, we are not given the Boolean function; but rather, we usually begin with a *verbal statement of a problem,* or at most, a truth table.

Two basic steps are involved in the determination of the simplest two-level logic circuit utilizing a given truth table:

1. Derive a Boolean function for the truth table in either sum-of-products or product-of-sums form.
2. Simplify this function.

MINTERM AND MAXTERMS

Consider the following two-variable Boolean algebra expression: $F = \bar{A}B + A\bar{B}$. This form of Boolean expression is referred to as the "Sum of the products" (minterm) as each term contains all the variables whether they are inverted or not inverted and then all terms are OR'ed together, (the addition sign represents the OR operator).

A second basic form of Boolean expression is the "Product of sums" (maxterm): $F = (A + B)(\bar{A} + \bar{B})$. Again each term contains all variables (inverted or not inverted) and each term then "anded" with the other terms (the multiplication sign represents the "AND" operator).

PRODUCT TERMS AND MAXTERMS

The maximum number of product terms, or maxterms, increases very rapidly with the increase in the number of variables since the maximum number of product terms is equal to 2^n (two raised to a power equal to the number n, of variables). In the previous example, we have two variables, so the maximum number of product terms is $2^n = 2^2 = 4$. This number of terms is obtained by assigning the two possible values 0 and 1, to each variable and combining these variables in all possible forms.

TRUTH TABLES

The study and derivation of logic expressions is facilitated by employing truth tables instead of multiplying each variable and its complement by the other variables.

A truth table is a simplified manner to obtain and present the maximum number of product terms and to evaluate the expression for each of the possible combinations of the variables in that expression. Then simplification can be attempted.

How a Truth Table is Made

Consider the expression A + BC. There are three variables in this expression, each of which can have the values 0 or 1. The possible combination of values may be arranged in binary ascending order, as shown in Table 13. Notice in Table 13 that the variables in each row of this table may be combined to form a binary number, from 000 to 111 (from 0 to 7 in decimal numbers).

Table 13		
A	B	C
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

The table is formed by writing ones and zeros alternately down the first column at the right, writing ones and zeros in series of two down the second or center column, in groups of four down the third column, etc. For additional variables, double the number of ones or zeros written in each group.

In the expression A or $B\bar{C}$, one of the variables, C, is complemented, so a column is now added to Table 13, listing values of C. This has been done in Table 14.

Table 14			
A	A	C	$\bar{C}$
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Note in Table 14 that whenever C is one, $\bar{C}$ is 0 and vice versa.

A column is now added (Table 15) listing the values $B\bar{C}$ assumes for each value of B and $\bar{C}$. This column will contain the value 1 only when both B and $\bar{C}$ are equal to 1.

Table 15				
A	B	C	$\bar{C}$	$B\bar{C}$
0	0	0	1	0
0	0	1	0	0
0	1	0	1	1
0	1	1	0	0
1	0	0	1	0
1	0	1	0	0
1	1	0	1	1
1	1	1	0	0

Now the logic addition of the values of "A" to the values which have been calculated for $B\bar{C}$ is performed in a final column (Table 16). This column will contain the value 1 only when "A" is equal to 1, when $B\bar{C}$ is equal to 1, or when both are equal to 1. The value of the expression $A + B\bar{C}$ could have been derived from Table 13, considering that $\bar{C}$ is 1 when C is 0 and vice versa. However, in the evaluation of more complex expressions as those containing parentheses and multiplications, as for example $(A + B)(\bar{A} + \bar{B})$ it is easier to evaluate the complete expression, term by term, as explained above.

Table 16					
A	B	C	$\bar{C}$	$B\bar{C}$	$A+B\bar{C}$
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	1	1	1
0	1	1	0	0	0
1	0	0	1	0	1
1	0	1	0	0	1
1	1	0	1	1	1
1	1	1	0	0	1

DERIVATION OF BOOLEAN EXPRESSIONS

When designing a logic circuit, the designer works from two sets of known values: 1) the various states which the inputs to the logic network can take, and 2) the desired outputs for each input condition. The logic expression is derived from these sets of values.

Consider a specific problem. A logic network has three inputs: A, B and C, and an output Z. The relationship between inputs and outputs is to be as follows: It is now necessary to add another column to the table. This column will consist of a list of "product terms" obtained from the values of the input variables. The new column will contain each of the input variables listed in each new row of the table, with the letter representing the respective input complemented when the input value for this variable is 0, and not complemented when the input value for this variable is 0, and not complemented when the input value is 1. The terms obtained in this manner are the product terms of the variables.

Table 17			
Inputs			Output
A	B	C	Z
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

With three input variables, A, B and C, each row of the table will contain a product term consisting of A, B and C with A, B or C complemented or not depending on the input values for that row. Since the number of variables is 3, the number of product terms is $2^n = 2^3 = 8$. Whenever Z is equal to 1, the A, B, and C product term in the same horizontal row is removed and formed into a "sum-of-products." Therefore, the product terms in the first, third, fifth and seventh horizontal rows are written as follows:

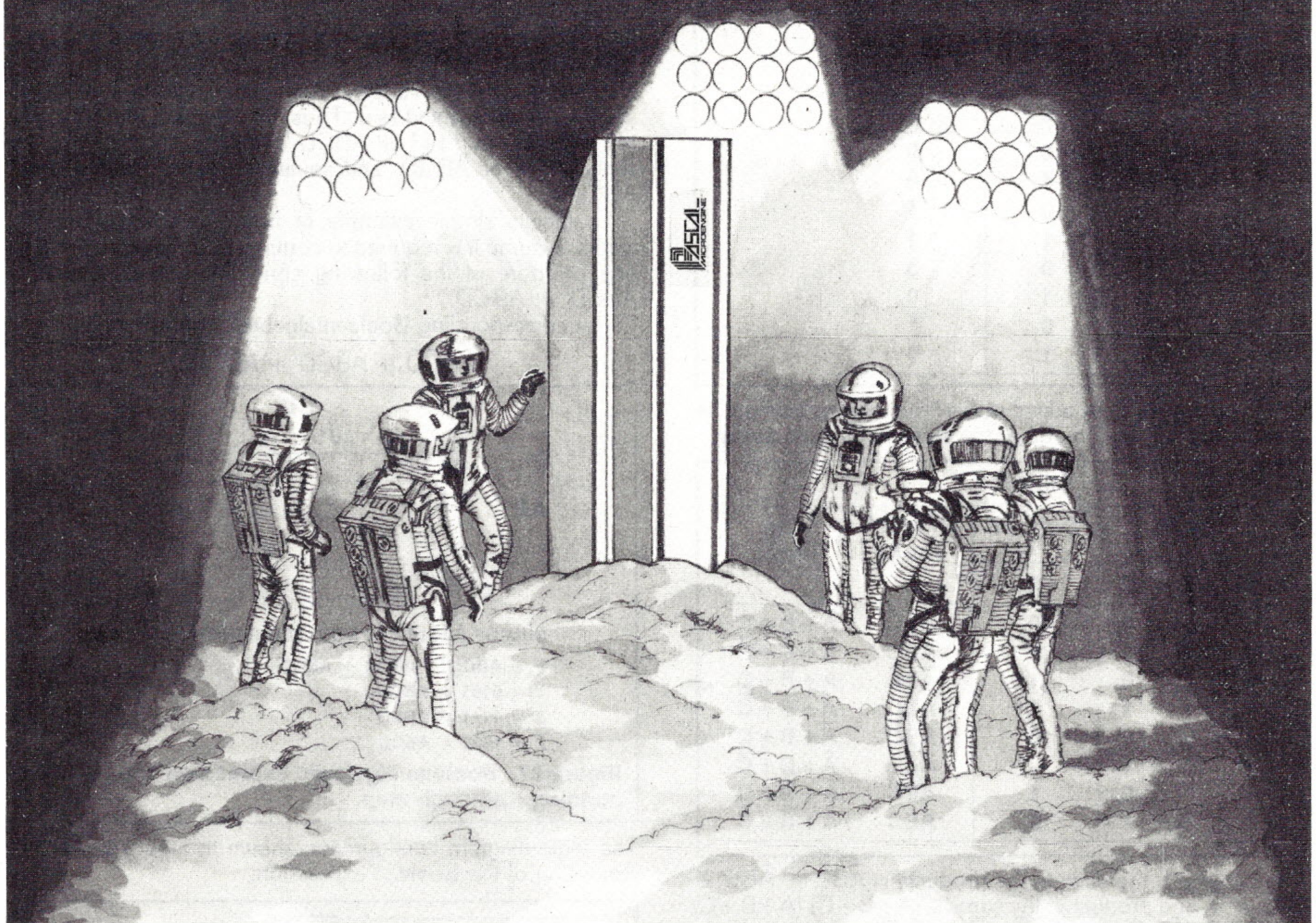
$$\bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + A\bar{B}\bar{C} + AB\bar{C}.$$

Table 18				
Inputs			Output	Product Term
A	B	C	Z	
0	0	0	1	$\bar{A}\bar{B}\bar{C}$
0	0	1	0	$\bar{A}\bar{B}C$
0	1	0	1	$\bar{A}B\bar{C}$
0	1	1	0	$\bar{A}BC$
1	0	0	1	$A\bar{B}\bar{C}$
1	0	1	0	$A\bar{B}C$
1	1	0	1	$AB\bar{C}$
1	1	1	0	ABC

There are now four sets of terms, each the product of three variables. The sum of these products is equal to the expression desired. This type of expression is often referred to as a canonical expansion for the transmission function. The complete expression in normal form is:

$$\bar{A}\bar{B}\bar{C} + \bar{A}B\bar{C} + A\bar{B}\bar{C} + AB\bar{C} = Z.$$

THE PASCAL ODYSSEY



WHAT'S SO GREAT ABOUT PASCAL?

The Pascal programming language is a *masterpiece* of human creativity. Originated by Niklaus Wirth at the Swiss Institute of Technology in 1968, Pascal was intended to aid in the teaching of *programming* as a *systematic discipline,* where the fundamental concepts of algorithms could be clearly and naturally expressed.

Pascal makes it easy for the user to implement virtually *any* algorithm *reliably*, at an abstract, human level, rather than having to perform the usual mind-bending contortions required by most other programming languages. The elegance of the language has resulted in an unprecedented *wave of enthusiasm* among Earth's knowledgeable computer users, toward adopting Pascal as an *international standard programming language.*

WHAT'S A MICROENGINE?

The **Pascal microengine™** is the world's first production implementation of Pascal in *hardware.* It executes UCSD Pascal intermediate code (P-code) *directly* as its *machine language*, thus being the first true "P-machine". The significance of this accomplishment should not be overlooked!

The desktop computer includes the 16-bit **Pascal microengine** processor, 64K bytes of RAM memory, complete DMA control functions, fully-integrated floppy disk controller (switch selectable for single or double density, mini or 8" floppy, and 1 to 4 disk drives), two RS-232 serial ports, two 8-bit parallel ports, floating point hardware, built-in power supply, self-test microdiagnostics, and an ASCII console.

The unit includes the complete UCSD Pascal operating system: Pascal compiler, BASIC compiler, file manager, screen-oriented text editor, program debugger, and a graphics package, all supported by thorough documentation. UCSD's version of Pascal incorporates random access file capability in addition to the sequential access capability of the original Pascal, and also includes x-y graphics.

For most business, educational, and personal applications, UCSD Pascal is a nearly ideal implementation tool. There are more users of UCSD Pascal today than users of all other versions combined. UCSD and Western Digital have agreed mutually to support this LSI implementation as the true UCSD standard.

HOW MUCH?

Khalsa Computer Systems offers the **Pascal microengine™** alone at $2995, with peripherals available at extra cost. A typical *complete* system, included CRT, 80-column printer, and dual 8" floppy disk drives costs less than $7000.

For those of you who want to build the *brilliant machines* of the future, the **Pascal microengine™** chipset is also available separately for $195.

We offer significant quantity discounts on all the above items. Please write or call for more information.

The Journey Begins At

KHALSA

COMPUTER SYSTEMS INCORPORATED

500 South Lake Avenue, Pasadena, CA 91101 (213) 684-3311
Tuesday-Friday 12 noon to 8 pm, Saturday 12 noon to 6 pm
Closed Sunday and Monday

DERIVATION OF A THREE-INPUT-VARIABLE EXPRESSION

Consider the following table expressing an input-to-output relationship for which an expression is to be derived:

Table 19

Inputs			Output
A	B	C	Z
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

Two columns will be added at this time, one containing the sum-of-products and the other the product-of-sums terms.

Table 20

Inputs			Outputs	Product Terms	Sum Terms
A	B	C	Z		
0	0	0	0	$\bar{A}\bar{B}\bar{C}$	$A+B+C$
0	0	1	0	$\bar{A}\bar{B}C$	$A+B+\bar{C}$
0	1	0	1	$\bar{A}B\bar{C}$	$A+\bar{B}+C$
0	1	1	1	$\bar{A}BC$	$A+\bar{B}+\bar{C}$
1	0	0	0	$A\bar{B}\bar{C}$	$\bar{A}+B+C$
1	0	1	0	$A\bar{B}C$	$\bar{A}+B+\bar{C}$
1	1	0	1	$AB\bar{C}$	$\bar{A}+\bar{B}+C$
1	1	1	0	ABC	$\bar{A}+\bar{B}+\bar{C}$

In Table 20, the sum of the products is $\bar{A}B\bar{C} + \bar{A}BC + ABC = Z$, and product of the sums is $(A+B+C)\,(A+B+\bar{C})\,(\bar{A}+B+C)\,(\bar{A}+B+\bar{C})\,(\bar{A}+\bar{B}+\bar{C}) = Z$. The shortest sum of products expression obtained is $\bar{A}B+B\bar{C}=Z$, whereas the shortest product of sums is $B(\bar{A}+\bar{C})$.

In some cases, the minimal sum-of-products expression will require fewer logic elements to construct; and in other instances the construction of the minimal product-of-sums expression will require fewer elements. If the sole criterion is the number of logic elements, it is necessary to obtain both a minimal sum-of-products expression and also a minimal product-of-sums expression in order to compare the two.

Referring to Table 20, we can also write the equation in its false output form:

$$\bar{Z} = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}C + A\bar{B}\bar{C} + A\bar{B}C + ABC$$

It is quite obvious the true output equation is the simplest to work with. To obtain the true output maxterm, invert the false output minterm.

KARNAUGH MAPS

To make a Karnaugh map, a Boolean function must be expressed in a "sum of the products" form. Consider the following four-variable expression: $F = ABCD + \bar{A}B\bar{C}D + A\bar{B}\bar{C}\bar{D}$. A four-variable Karnaugh map suitable for the problem is shown in Figure 86. The variables are A, B, C, and D. Since each variable can represent two values, namely 0 and 1, there are sixteen combinations of ABCD from 0000 to 1111. The Karnaugh map is composed of sixteen squares, one for each possibility.

AB \ CD	00	01	11	10
00	$\bar{A}\bar{B}\bar{C}\bar{D}$	$\bar{A}\bar{B}\bar{C}D$	$\bar{A}\bar{B}CD$	$\bar{A}\bar{B}C\bar{D}$
01	$\bar{A}B\bar{C}\bar{D}$	$\bar{A}B\bar{C}D$	$\bar{A}BCD$	$\bar{A}BC\bar{D}$
11	$AB\bar{C}\bar{D}$	$AB\bar{C}D$	$ABCD$	$ABC\bar{D}$
10	$A\bar{B}\bar{C}\bar{D}$	$A\bar{B}\bar{C}D$	$A\bar{B}CD$	$A\bar{B}C\bar{D}$

Figure 86.

The presence of a variable, A is represented as 1 while its absence is represented as 0. Thus, the square X in Figure 87 is interpreted as 1111 or ABCD. Similarly, square Y represents 0101 or $\bar{A}B\bar{C}D$, and square Z represents 1000 or $A\bar{B}\bar{C}\bar{D}$.

To take a simple example of how this Karnaugh map works, assume it is required to complete, or make, a circuit if one or more of the following conditions prevail: $AB\bar{C}D$, $A\bar{B}\bar{C}D$, or $A\bar{B}\bar{C}\bar{D}$.

The corresponding Boolean algebraic expression is:

$$A\bar{B}\bar{C}\bar{D} + A\bar{B}\bar{C}D + AB\bar{C}D.$$

AB \ CD	00	01	11	10
00				
01		Y		
11			X	
10	Z			

SUM OF PRODUCTS BOOLEAN FUNCTION

F = ABCD + ABCD + ABCD

Y = 0101 = ABCD

X = 1111 = ABCD

Z = 1000 = ABCD

Figure 87. Boolean algebraic expression and corresponding Karnaugh map.

The logic diagram configuration shown in Figure 88 is the realization of the Boolean expression.

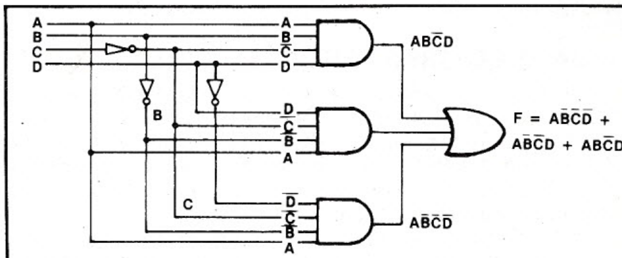
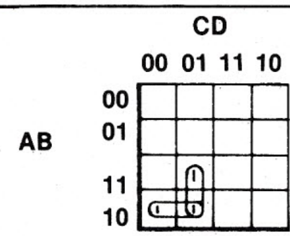


Figure 88. Logic diagram configuration of Boolean expression.

The following is the manner in which the simplification is accomplished through the use of the Karnaugh map shown in Figure 89.



1000 1001 1101

$F = A\bar{B}\bar{C}\bar{D} + A\bar{B}\bar{C}D + AB\bar{C}D$

$F = A\bar{B}\bar{C} + A\bar{C}D = A\bar{C}(\bar{B} + D)$

Figure 89. The simplified Boolean expression.

To obtain a completed circuit for the conditions stated in the Boolean expression under discussion, start by placing a 1 in the three squares corresponding to the three terms in the Boolean expression. Next, adjacent squares are grouped.

Two groups of two squares are possible, indicating that the preliminary Boolean expression containing two terms is possible. Overlapping is permissible.

With the horizontal grouping we see that, in the terms which correspond to this horizontal grouping, D can be either 0 or 1, so that the switching is independent of the state of D. Thus D can be eliminated. The result is a term $A\bar{B}\bar{C}$ instead of $A\bar{B}\bar{C}\bar{D} + A\bar{B}\bar{C}D$.

With the vertical grouping, formed by terms $ABCD$ + $AB\bar{C}D$, B can be eliminated. The resulting term is $A\bar{C}D$. The simplified Boolean expression then is $A\bar{B}\bar{C} + A\bar{C}D$.

With factoring, the expression becomes: $A\bar{C}(\bar{B} + C)$.

Accordingly, the simplified logic circuit can be drawn readily, as illustrated in Figure 90. This circuit will provide the desired switching, and it involves fewer gates than the circuit of Figure 88.

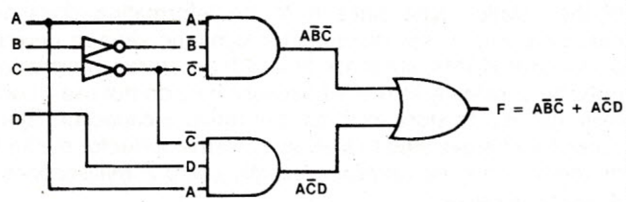


Figure 90. The simplified logic circuit.

The following is a brief example of a more complex expression.

$$F = \bar{A}\bar{B}\bar{C}\bar{D} + \bar{A}B\bar{C}\bar{D} + AB\bar{C}\bar{D} + A\bar{B}\bar{C}\bar{D} + \bar{A}\bar{B}\bar{C}D$$
$$\bar{A}\bar{B}CD + \bar{A}\bar{B}C\bar{D} + \bar{A}BCD + \bar{A}BC\bar{D} + ABCD$$
$$ABC\bar{D}$$

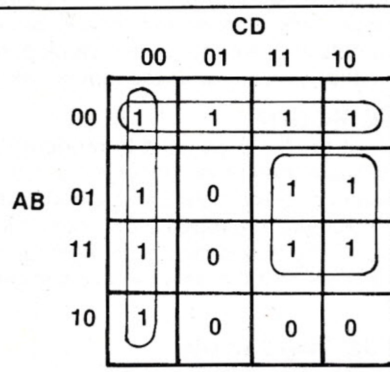


Figure 91.

Squares must always be combined so that a group will contain the greatest possible number of squares. However, groups size is limited to one, two, four, eight, and 16 squares. Three groupings of four adjacent squares are possible as shown.

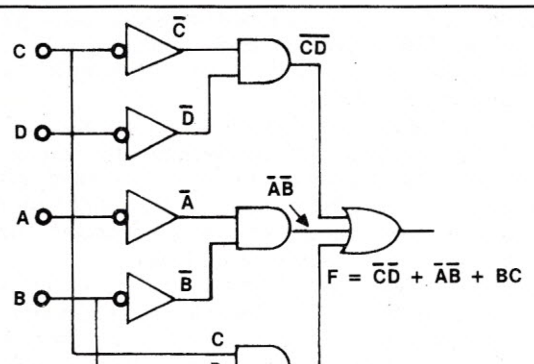


Figure 92. Equivalent logic diagram.

As can be seen by observing the Karnaugh Map at A in Figure 91, A and B can be removed from all the expressions ending in $\bar{C}\bar{D}$; C and D can be removed from all expressions starting with $\bar{A}\bar{B}$. And A and D can be eliminated from all expressions containing BC. The simplified Boolean expression is: $\bar{C}\bar{D}$ + AB + BC.

The equivalent logic diagram is illustrated in Figure 92.

QUALITIES OF BOOLEAN ALGEBRA

Boolean Algebra is a form of mathematics that is applicable to situations in which "yes" or "no" answers are obtained from a set of conditions, each of which can be either "true" or "false." Thus, Boolean Algebra lends itself admirably to switching circuits where the inputs can only be either "on" or "off," and each output can be either "on" or "off."

The Boolean algebra possesses several outstanding qualities which may be enumerated as follows:

1. Since it is mathematical in nature, it is an exact science which leaves nothing to chance.
2. Its expressions can be shorthand notations of both the definition of the problem and its solution.
3. The mathematical expression can be converted to hardware by inspection. Thus it is one of the few systematic approaches to circuit synthesis.
4. It provides a means of minimizing the amount of hardware required to perform a switching function.

The intent here has been not to treat simplification in an in-depth manner, but to show that there are pencil and paper methods to utilize, if one so desires.

LOGIC CIRCUITS

There are two basic classifications of logic elements:

1. Circuits whose output at a given time are dependent upon the conditions of the inputs at that given time are referred to as combinatorial circuits.
2. Circuits whose output is dependent upon the inputs at a point in time as well as the condition of the memory elements prior to receiving the inputs are referred to as sequential circuits. AND, OR and NOT gates are examples of combinatorial circuits.

In working with digital circuitry, we are dealing with steady d-c levels as well as pulses of d-c. The various logic elements are triggered (enabled) either by one or the other, or both. Figure 93 displays terminology associated with positive and negative going voltage transitions (changes).

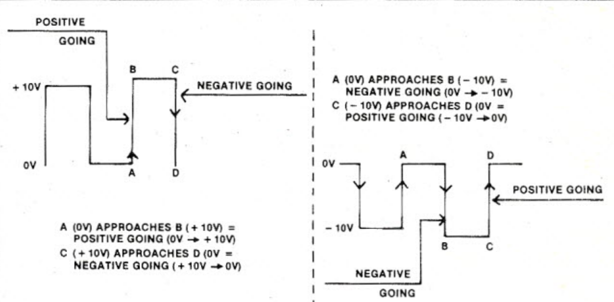


Figure 93. Positive and Negative Going Transitions.

Logic Symbols

Although there have been attempts to standardize the manner of logic element representation in drawings, some manufacturers either use their own symbols or modified versions of the industrial control standards and military standard.

MIL-STD-806B

A small circle at the input of a logic element indicates that the relatively low (L) input signal activates the function. The absence of a small circle indicates that the relatively high (H) input signal activates the function.

A small circle at the output of a logic element indicates that the output of the activated function is relatively low. The absence of a small circle indicates that the output of the activated function is relatively high.

Quite often one runs into indicators like those shown in Figure 94. A word of caution should be exercised here; ascertain how the manufacturer is using these flags.

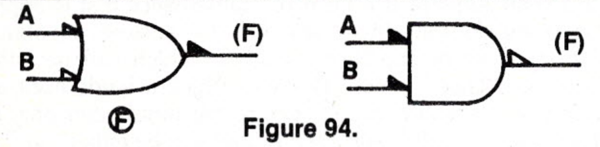


Figure 94.

LOGIC POLARITIES

If the more positive or less negative potential (voltage) is considered to be a logic one (1), we have positive logic. If the more negative or less positive potential (voltage) is considered to be a logic one (1), we have negative logic. In both cases a logic one is considered to be a high (H) and a logic zero (0) considered a low (L).

TABLE OF COMBINATIONS

Table 21 shows the logical equivalency that can prevail between two logic elements. For instance, considering positive logic, inverting the input and output of an OR gate produces the same output as that of an AND gate. Remember for positive logic a one is high and a zero is low.

Table 21

This table of combinations illustrates the applications and functions of two variables and equivalents.

RELATIVE LOGIC LEVEL MAGNITUDE	POSITIVE LOGIC DIGIT	NEGATIVE LOGIC DIGIT
HIGH	1	0
LOW	0	1

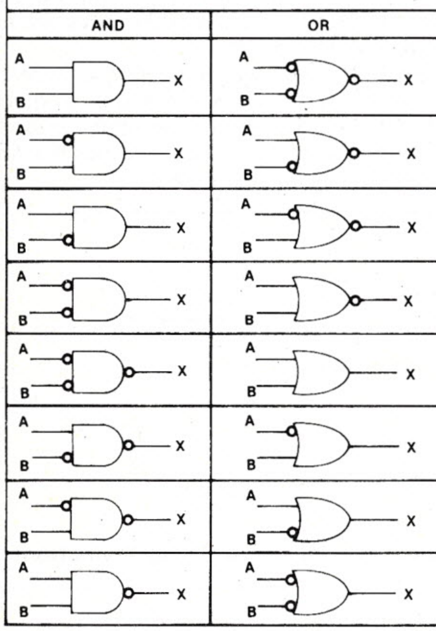


TABLE OF COMBINATIONS

A	B	X
H H L L	H L H L	H L L L
H H L L	H L H L	L L H L
H H L L	H L H L	L H L L
H H L L	H L H L	L L L H
H H L L	H L H L	H H H L
H H L L	H L H L	H L H H
H H L L	H L H L	H H L H
H H L L	H L H L	L H H H

FLIP-FLOPS

The logic circuits (gates) that have been studied all have one thing in common; their outputs at a given time are functions only of their inputs at that time. Circuits of this type are called "combinational logic circuits."

There is another basic function that must be performed in digital computers, which is the retention of information until it is to be used. Circuits of this type are called "sequential logic circuits."

Memories and Registers

In addition to the large-scale memory in which the data of the problem and the code of the operation to be executed are held, other computer units must also incorporate devices that retain information. Thus, the arithmetic unit must contain registers that receive information from the large-scale memory and retain it until the arithmetical operations have been completed; and the control unit must be able to hold each instruction of the code until it has been executed.

Bistable Characteristic

Generally, these storage functions are performed by high-speed circuits that have two distinct stable physical states (bistable elements). The bistable element represents one bit of information for each of its two stable states.

The bistable multivibrator (Eccles-Jordan) or flip-flop is often used as a memory device, which is considered as being of the "static" type since in it the information does not change position. (Semiconductor dynamic storage devices or dynamic RAMs, are those in which the stored information must be constantly refreshed because they do not use a complete flip-flop to store each bit, but rather a capacitor. Since capacitor charges tend to leak away, semiconductor dynamic memories must be refreshed usually every 2 milliseconds.)

Toggle Action

The flip-flop is also known as a "toggle" since its operation is very similar to that of a mechanical toggle switch, it is either on or off. We call these two stable states "set" and "reset."

The flip-flop, depending upon its logic function, is provided with various synchronous or asynchronous controls to set the output to the desired state.

Set (S or Sd) and Reset (R or Rd) inputs are direct or asynchronous controls. Other inputs such as J, K, Sc, and Rc, are synchronous inputs and depend upon a clock pulse to transfer logic information present on these inputs into the device.

FLIP-FLOP CIRCUITS

The majority of the flip-flops used in modern digital computers and microcomputers are incorporated in integrated circuits. However, in some equipment we still find flip-flop circuits using individual transistors. Also, since the operating principle of the IC flip-flop is similar to those using discrete units, we will analyze first the operation of transistor flip-flop circuits.

Transistor Flip-Flop Circuits

In a flip-flop circuit, under steady state conditions, one transistor is always in the "on" state while the other is in the "off" state. Flip-flop circuits may be divided into two major groups; circuits designed with transistor switches saturated and circuits designed with transistor switches non-saturated.

Saturated switches generally are not as fast as flip-flops using transistors that do not saturate, because storage time of minority carriers in a saturated transistor adds additional delay time. With a given dissipation rating for a transistor, a saturated flip-flop can switch more current than one using non-saturated transistors.

Also, for a given current level, there is less average dissipation in the saturated flip-flop because the two operating points have low dissipation. On the other hand, the "on" point of a non-saturated transistor is a point of high dissipation and represents a larger amount of lost power. Therefore, the switched current must be kept lower.

Even so, a flip-flop using non-saturated transistors of higher dissipation can be used to switch the same amount of load power at a faster rate than a saturated flip-flop. Since the transistor is never saturated, storage time is low; however, there is more lost power in the transistor. Lower power efficiency is the price for speed.

NOTE: A switching transistor that is conducting heavily (maximum current) and whose collector to emitter voltage is close to zero volts, is generally considered to be "saturated."

A switching transistor that is not conducting current and whose collector to emitter voltage is at or near VCC (supply voltage), is generally considered to be "cut off."

A switching transistor that is operating between the two extremes is considered to be operating below cut-off and above saturation.

Saturated Flip-Flop

The basic flip-flop circuit using saturated transistors is shown in Figure 95. This circuit which is called reset-set (R-S) flip-flop or latch is stable in either of two states. To change from one state to the other, signals must be applied alternately at either the A points for base triggering or the B points for collector triggering. Either type of triggering requires pulses from two sources. One is the set (S) pulse and the other the reset (R) pulse.

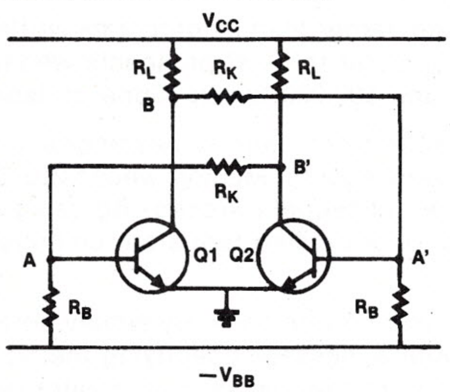


Figure 95. Basic Flip-Flop Transistor Circuit.

Non-Saturated Flip-Flop

One technique for designing a non-saturated flip-flop circuit is employing a Baker clamp to hold the transistors out of saturation, using a silicon diode as a reference diode and a germanium diode as a switching diode. This technique, applied to flip-flop circuits, is shown in Figure 96 where clamping reference diodes Dc have been added to the basic circuit of Figure 95.

During the time that the transistor is "on," diode DC clamps the collector voltage above the base voltage by the reference voltage of Db, thus keeping the transistor out of saturation. Diode Db is passing more than enough current to control the collector current and voltage. When the transistor is "off," Dc does not conduct and Db still maintains a reference voltage, but now passes only a very small current.

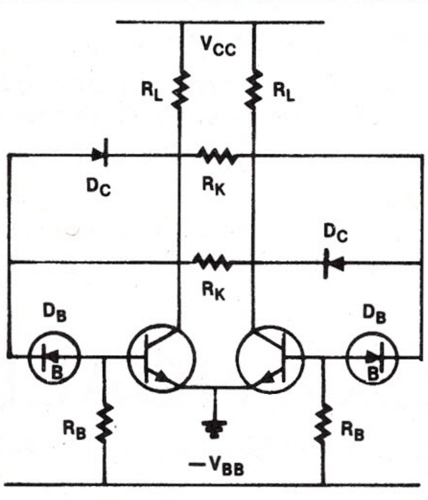


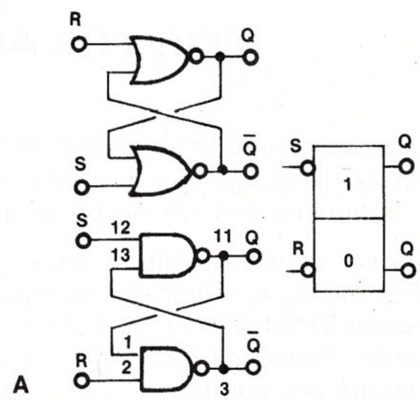
Figure 96. Basic Non-Saturated Flip-Flop.

FLIP-FLOPS USING IC LOGIC GATES

The R-S flip-flop of Figure 96, also called "latch," as well as many other types of flip-flops, can be implemented using IC NOR or NAND gates, as shown in A and B of Figure 97.

Latch

The R-S flip-flop or latch operates in the asynchronous mode (no clock pulse required). The NOR or NAND gates are connected in such a manner that the output of each gate provides an input to the other gate. The latch is controlled by two input steering lines called set (S) and reset (R).



Flip-Flop in "1" Condition Initially

	1	2	3	4	5	6
	INITIAL CONDITIONS		PULSED SIGNAL INPUTS		OUTPUT RESULT	
	1 OUTPUT	0 OUTPUT	SET	RESET	1 OUTPUT	0 OUTPUT
1	HI	LO	LO	LO	INDETERMINATE	
2	HI	LO	LO	HI	HI	LO
3	HI	LO	HI	LO	LO	HI
4	HI	LO	HI	HI	HI	LO
5	LO	HI	LO	LO	INDETERMINATE	
6	LO	HI	LO	HI	HI	LO
7	LO	HI	HI	LO	LO	HI
8	LO	HI	HI	HI	LO	HI

Flip-Flop in "0" Condition Initially

B TRUTH TABLE

Figure 97. Basic Latch (R-S) Flip-Flop, symbol and truth table.

Clocked R-S Flip-Flop

Figure 98 shows the arrangement, logical symbol and truth table for the clocked R-S flip-flop. This circuit has two separate sections with the flip-flop section being the same as in the R-S flip-flop shown in B of Figure 97.

D-Type Flip-Flop

Figure 99 shows the logical symbol and a truth table for the D-type (Data type) flip-flop. This flip-flop requires only

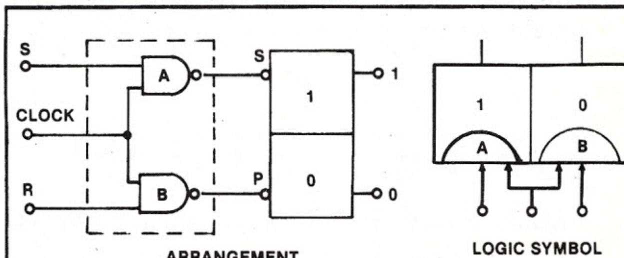


Figure 98a.

INITIAL CONDITIONS		SIGNAL INPUTS		AFTER CLOCK PULSE	
1 OUTPUT	0 OUTPUT	SET	RESET	1 OUTPUT	0 OUTPUT
LO	HI	LO	LO	LO	HI
LO	HI	LO	HI	LO	HI
LO	HI	HI	LO	HI	LO
LO	HI	HI	HI	INDETERMINATE	
HI	LO	LO	LO	HI	LO
HI	LO	LO	HI	LO	HI
HI	LO	HI	LO	HI	LO
HI	LO	HI	HI	INDETERMINATE	

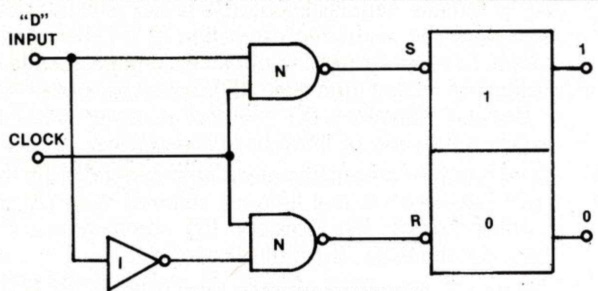
TRUTH TABLE

Figure 98b. Clocked R-S Flip-Flop.

one conditioning input, which can be either high or low and thus insures that there can be no indeterminate state in its operation.

The single conditioning input is called "D" or "Data" input; and whatever information is present at the D input prior to and during the clock pulse propagates to the "1" output when the leading edge of the clock pulse occurs.

Should the D input be high prior to and during a clock pulse, the flip-flop goes to the "1" state; if the D input should be low prior to and during a clock pulse, the flip-flop will go to the "0" state.



(A) LOGICAL SYMBOL

INITIAL CONDITIONS		D INPUT	AFTER CLOCK PULSE	
1 OUTPUT	0 OUTPUT		1 OUTPUT	0 OUTPUT
LO	HI	LO	LO	HI
HI	LO	LO	LO	HI
LO	HI	HI	HI	LO
HI	LO	HI	HI	LO

(B) TRUTH TABLE

Figure 99. D-Type Flip-Flop.

Master-Slave J-K Flip-Flop

The J-K flip-flop (Figure 100) is one of the most useful flip-flops in use because of the following features:

A clock pulse will not cause any transitions in the flip-flop if neither the J nor the K input is enabled prior to the application of the clock pulse.

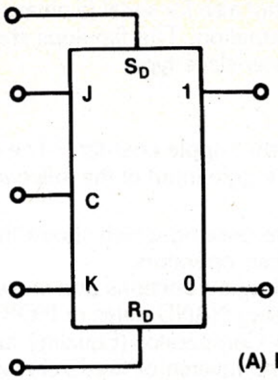
If both the J and the K inputs are enabled before application of a clock pulse, the flip-flop will complement, or change state, when the clock pulse occurs.

If a 1 is present before the clock pulse, a 0 will be present after the clock pulse. If 0 is present before the clock pulse, a 1 will be present after the clock pulse.

Application in ICs

The J-K flip-flops used in integrated circuits are master-slave types which trigger on the trailing edge of a clock pulse. The information present at the J and K inputs is transmitted to the master flip-flop on the positive-going leading edge of a high clock pulse, and there until the negative-going pulse, the flip-flop will complement, or change state, when the clock pulse occurs.

If a 1 is present before the clock pulse, a 0 will be present after the clock pulse. If a 0 is present before the clock pulse, a 1 will be present after the clock pulse.



(A) LOGICAL SYMBOL

INITIAL CONDITIONS				FINAL CONDITIONS	
OUTPUT		INPUTS		OUTPUT	OUTPUT
1	0	J	K	1	0
LO	HI	LO	LO	LO	HI
LO	HI	LO	HI	LO	HI
LO	HI	HI	LO	HI	LO
LO	HI	HI	HI	HI	LO
HI	LO	LO	LO	HI	LO
HI	LO	LO	HI	LO	HI
HI	LO	HI	LO	HI	LO
HI	LO	HI	HI	LO	HI

(B) TRUTH TABLE

Figure 100. Master-Slave J-K Flip-Flop.

FF — APPLICATIONS

Register

A group of flip-flops so connected for the storage of binary information (computer word or byte). Registers are not only

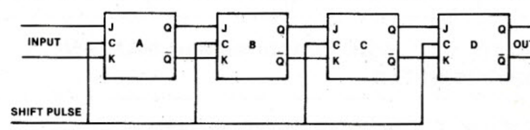
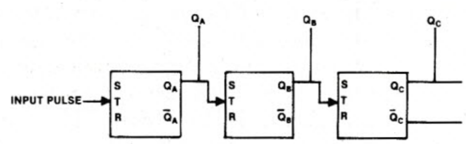


Figure 101a. 4-Bit Shift Register



Q_C	Q_B	Q_A	PULSE COUNT
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

Figure 101b. Binary Ripple Counter

used for temporary storage but also for serial (a binary bit at a time) transfer of binary information. The flip-flops shown in Figure 101a are of the master-slave type.

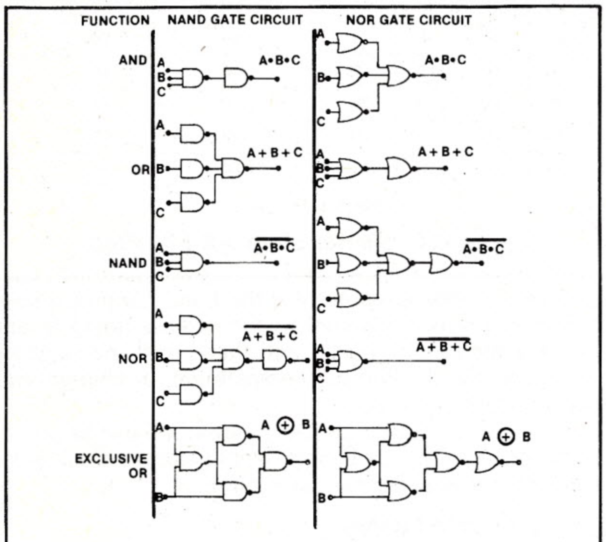
Counters

Figure 101b shows a binary "ripple counter." The output of one flip-flop is tied to the toggle input of the following flip-flop.

Figure 102 is a composite drawing which shows the relations of the five basic Boolean operators.

Figure 102 presents the Logic Functions and shows how they can be implemented using NAND gates or NOR gates. The implementation for the Comparator (Equality) function is not shown because it is the inverse of the Exclusive-OR.

Note in Figure 102 that to obtain an AND gate using NOR gates, four of these gates are needed. On the other hand, to obtain an OR gate using NAND gates four NAND gates are needed. Also, to implement a NAND gate using NOR gates, five of these gates are needed; and to implement a NOR gate using NAND gates five of these gates are needed.



INPUTS		OUTPUTS					
A	B	AND $A \cdot B$	NAND $\overline{A \cdot B}$	OR $A+B$	NOR $\overline{A+B}$	EXCL.-OR $A \oplus B$	EQ.COMP. $\overline{A \oplus B}$
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1

Figure 102. Logic Functions and NAND/NOR Implementation.

INTRODUCTION TO LOGIC FAMILIES

Vast changes in digital equipment and technologies have taken place since the early 1950 Univac computer. These changes have been brought about by the continuing developments of semiconductor techniques. Among the first in the IC logic family was RTL (Resistor Transistor Logic) and Diode Transistor Logic (DTL). Then TTL (Transistor Transistor Logic) made its appearance and is still here in a wide variety of components.

After the advent of the discrete FET came Metal Oxide Semiconductor (MOS) and Complimentary Metal Oxide Semiconductor — CMOS. Among some of the newer types are I^2L (Integrated Injection Logic) and Tri-state Logic. Why so many types? Costs and technical trade offs naturally, per application. Among these trade offs are:

1. Propagation time (speed)
2. Power dissipation
3. Noise margin
4. Fan in and Fan out capabilities
5. Last but not least, packaging density.

Classification of packaging densities are:

1. Small scale integration (SSI)
2. Medium scale integration (MSI)
3. Large scale integration (LSI)
4. Very large scale integration (VLSI)

NEXT MONTH

Next month, the individual logic families will be discussed in greater detail, prior to looking at the microprocessor as a system component.□

SUMMARY/QUIZ TUTORIAL #4

1. The flip-flop is also known as (A) a divider; (B) an alternate; (C) a sequential; (D) a twin; (E) a toggle.
2. The Data type flip-flop requires (A) only one conditioning input; (B) an even number of conditioning inputs; (C) an odd number of conditioning inputs but higher than five; (D) no conditioning input at all; (E) only four conditioning inputs.
3. A switching transistor whose collector to emitter voltage measures VCC is (A) saturated; (B) cut off; (C) operating at its maximum point of power; (D) conducting current heavily; (E) forward biased.
4. Circuits that include the retention of information until it is to be used are called (A) reserve circuits; (B) combinational logic circuits; (C) complementary circuits; (D) sequential logic circuits; (E) dual circuits.
5. A Karnaugh map (A) is a graphical device designed to aid in circuits simplification; (B) is not a convenient means for the visual representation of a 2-state function; (C) cannot be used to obtain a simplified Boolean expression from a truth table; (D) cannot be made from a Boolean function; (E) method is never used to reduce a function of three or four variables.
6. Circuits whose output at a given time depend upon the input conditions at that time are referred to as (A) sequential circuits; (B) registers; (C) combinatorial circuits; (D) flip-flops; (E) multivibrators.
7. One of the new types of logic developed is (A) DTL; (B) CMOS; (C) RTL; (D) I^2L; (E) MOS.
8. The R-S Latch (A) operates in a synchronous mode; (B) produces a truth table that is the same as a J-K flip-flop; (C) operates in the asynchronous mode; (D) is controlled by one steering line; (E) has five inputs.
9. Which of the following is not a combinatorial element (A) NOR; (B) AND; (C) OR; (D) R-S Latch; (E) NAND.
10. If the lower of two voltages is considered to be a logic high (A) positive logic is being considered; (B) then zero volts represents a false output; (C) we have an invalid operation; (D) negative logic is being considered; (E) the switching transistor is cut-off.
11. To make a Karnaugh map, a Boolean function must be expressed in (A) sum form; (B) sum of the product form; (C) product form; (D) product of sum form; (E) maxterm form.
12. The function (A+B) (A+B) (A) is expressed in maxterm; (B) is considered a minterm; (C) is a sum of the products form; (D) cannot be expressed in any other form; (E) cannot be simplified any further.
13. Sum-of-the-products (A) is the same as product-of-sums; (B) is an expression that cannot be derived using a table of input and output values; (C) is an expression often referred to as a canonical expansion for the transmission function; (D) is the solution of theorem; (E) are minterms.
14. Please rate the fourth unit of the NTS/INTERFACE AGE Mini Series. (A) Excellent; (B) Good; (C) Average; (D) Poor.

SPECIAL BONUS OFFER

FREE Applesoft Firmware card with the purchase of your 16K Apple II Computer!

on the very affordable apple® computer

We are offering the basic Apple II Computer with 16K of memory, PLUS the Applesoft Firmware Card—a $200 value— all for $1195. This offer is good for a limited time only: just place your order no later than August 15th.

Our special offer includes the Apple II with 16K of memory, 2 remote controller paddles, the free Applesoft Firmware Card, operating software, demonstration graphics software on cassette, and operating manuals. Our photograph also shows many of the accessories available, at additional cost, to expand your Apple II later.

Brilliant Color and High Resolution Graphics
Plus Communication and Sound Capabilities

EXPANDABLE:
- Single or dual floppy disk drives
- Printer
- Monitor (or use your own color TV)
- TV Modulator
- Memory to 48K
- Communications Interface
- Accoustic Coupler
- Voice Recognition unit

APPLICATIONS:
- Educational Programs and Games
- Dow Jones Stock Market Analysis
- Home finance

Ask us about our new Apple Business Software package.

Sunshine Computer Company

20710 South Leapwood Ave.
Carson, California 90746 (213) 327-2118

***FREE Applesoft Firmware Card**
A $200 VALUE

When you order a 16K Apple II Computer from Sunshine. Place your order today to receive your free bonus by filling out this coupon and enclosing your check or money order. But hurry! This free offer is valid ONLY until August 15, 1979.

Yes, I want to order my Apple II with free Applesoft Firmware Card today. Enclosed is my check (or money order) for $1195. I understand that delivery is 3-5 weeks. (California residents add 6% state sales tax.)

Name ____________________

Street ____________________

City __________ State ____ Zip ______

All orders shipped FOB Carson, California. Sunshine Computer Co. reserves the right to specify shipper.

Warning: Don't rely on reader service card . . . it'll arrive too late to take advantage of this free offer. Send in this order form, or duplicate on a plain piece of paper, to reserve your Apple II and free Applesoft Firmware Card.

Send to:
SUNSHINE COMPUTER COMPANY
20710 S. Leapwood Ave., Carson, CA 90746

NEW PRODUCTS

Business Computer with Turn-Key Applications Software

A desk-top computer that is also a word processor, an accounting system, an auto finance system and a business information system is being introduced by Compucorp.

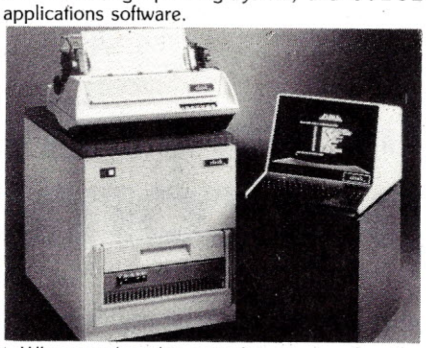
The Compucorp 625 Mark II is offered with an extended BASIC language operating system and up to 64 kbytes of internal memory.

For more information contact Compucorp, 1901 S. Bundy Dr., Los Angeles, CA 90025, (213) 820-2503.

CIRCLE INQUIRY NO. 121

Clerk™ 800 Computer System

The Retail Sciences, Inc. Clerk 800 features a 16-bit single board processor with 64K memory, a 32 megabyte cartridge disk storage subsystem, a multi-tasking operating system, and COBOL applications software.



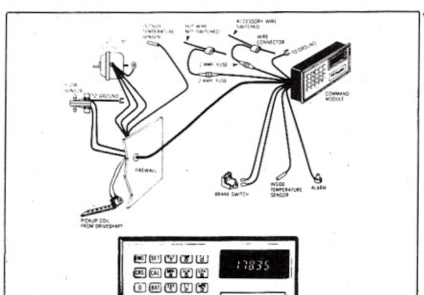
When combined in a single system such as the Clerk 800, these features mean system expandability at low cost.

For details contact Retail Sciences, Inc., Mktg. Dept., Suite 254, 3384 Peachtree Rd., N.E., Atlanta, GA 30326, (404) 231-2303.

CIRCLE INQUIRY NO. 122

Electronic Co-Pilot for Cars

CompuCruise, a new electronic co-pilot, can maintain your car, camper, or truck at the most fuel-efficient highway speeds. Besides electronically accurate cruise control, CompuCruise tells you instantly how much time (at current driving speed) and how far until the tank is empty; how much fuel has been used since the tank was filled; how much fuel the entire trip took.

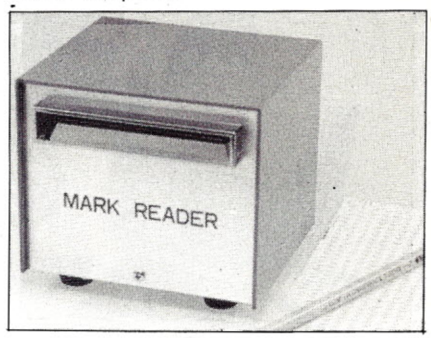


Price is $199.95. Contact Bits, Inc., P.O. Box 428, 25 Route 101 West, Peterborough, NH 03458. Order by phone at 800-258-5477.

CIRCLE INQUIRY NO. 123

Sense Card Reader

The MR-500 is a mark sense card reader specifically designed for the microprocessor computer market by Chatsworth Data Corporation. The MR-500 is a hand-fed reader that accepts cards of variable length marked with a standard number two pencil.



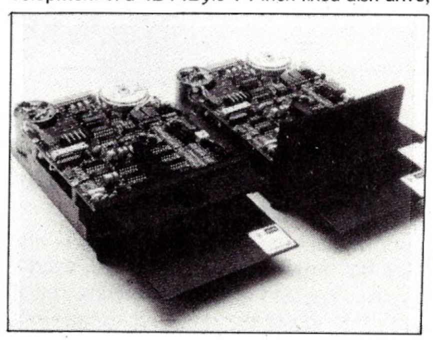
The card data is converted to either ASCII or card image. Special interfaces allow the reader to be connected directly to the TRS-80, Apple II, and PET computers.

Price is $750 with interface. Delivery is 90 days. For details contact Chatsworth Data Corp., 20710 Lassen St., Chatsworth, CA 91311.

CIRCLE INQUIRY NO. 127

210mm Fixed Disk Drive

BASF Systems is developing a 210mm (8¼-inch) fixed disk drive, based on technology pioneered at its West German laboratories. After development of a 42 MByte 14-inch fixed disk drive,



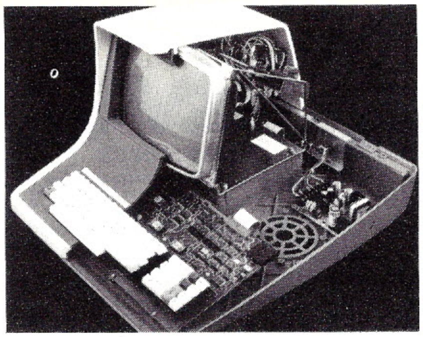
BASF AG shifted certain research on 3350 Disk/Head technology to its U.S. subsidiary. The new BASF drive incorporates one or more 210mm disks.

For more information contact BASF Systems, Computer Products, Crosby Dr., Bedford, MA 01730.

CIRCLE INQUIRY NO. 130

New Terminal from Intertec

The InterTube II video display terminal offers standard features that include an upper and lower case character set displayed on an 8x10 dot matrix; a full 24 line by 80 character screen; a status line which is displayed in half intensity; a complete ASCII keyboard with an 18-key numeric pad; 14 user-defined function keys; full cursor addressing; automatic repeat of all keys; individual backspace and shiftlock keys and a graphics mode to facilitate easy design and display of all types of forms.



Price is $995. OEM prices range from $798-$598 depending on quantity. For more information contact Intertec Data Systems Corp., 2300 Broad River Rd., Columbia, SC 29210, (803) 798-9100.

CIRCLE INQUIRY NO. 133

New Interface for TRS-80

H & K Computer Corporation recently announced the development of their new parallel to serial interface module for Radio Shack's TRS-80 microcomputer.

Use of the new module allows the TRS-80 user to expand his system without purchasing the expansion chassis.

The new parallel to serial interface is plug compatible with the TRS-80 keyboard, CRT or expansion chassis and other printers. Baud rates and line feeding are switch selectable with automatic buffering and delay for the carriage return; allowing compatibility with individual terminals.

Price is $150. For more information contact Harold Schonhoeft, President, H & K Computer Corp., 15 East 31st St., Kansas City, MO 64108, (816) 561-1776.

CIRCLE INQUIRY NO. 135

Exorciser CRT Controllers

The EXO-2480 Motorola Exorciser bus compatible alphanumeric CRT controller features Transparent Memory. This plug-in board provides 96 ASCII characters with descenders plus 32 graphic symbols in one page 24x80 or two pages 24x40.

Normal or inverse video, with or without blink synchronized internally or from an external source (TV camera or VTR) are some of the features available on the EXO-2480.

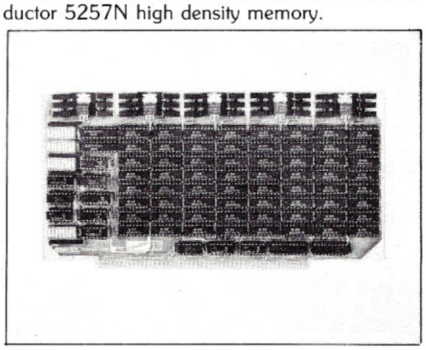
The EXO-512 may be used to provide a high resolution graphics, 256 vertical x 512 horizontal display or two independent 256x256 images. Four level grey scale is easily obtained by combining these images. Both boards may act as master or slaves in an alphanumerics/graphics application and by adding a second EXO-512, an image containing 4 bits pixel is produced.

Price for one EXO-2480 is $495; EXO-512, $695. For more information contact Matrox Electronic Systems Ltd., 2795 Bates Rd., Montreal, Quebec H3S 1B5, Canada.

CIRCLE INQUIRY NO. 136

Econo RAM XIII

Godbout Electronics is now offering a 32K RAM memory board that features bank select, phantom select, independent port decoding for each bank. The board uses National Semiconductor 5257N high density memory.



The board comes with a one year warranty. Price is $629 in kit form and $699 assembled. For more information contact Godbout Electronics, Box 2355, Oakland Airport, California 94614.

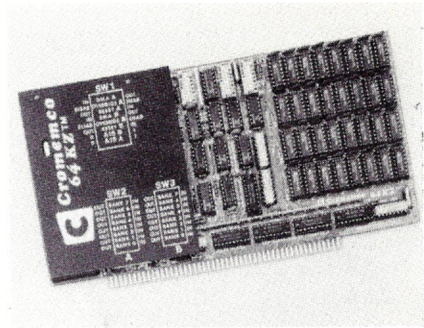
CIRCLE INQUIRY NO. 138

64KZ RAM Card

A new 64KZ Ram card is now available from Cromemco. The 64KZ is an S-100 bus compatible 65,536 byte read/write memory board.

The 64KZ incorporates 16K RAM chips with 150 nsec access times to achieve its high density and high speed operation. These 150 nsec RAM chips insure adequate timing margins for propagation delays. This means the 64KZ reliably operates in 4MHz Z-80 systems with absolutely no wait states.

The card includes automatic 64KZ enable or disable after a system RESET.

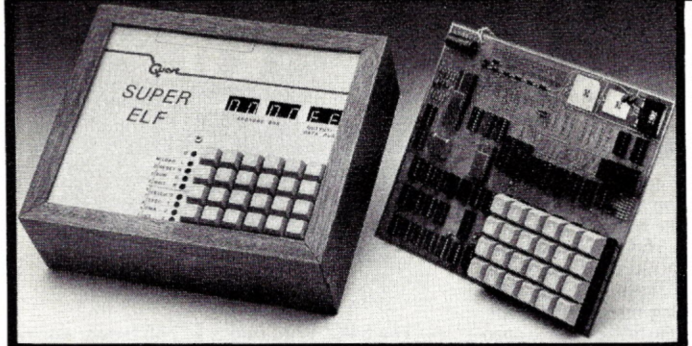


The 64KZ RAM card is $1785 factory assembled and tested. For details contact Cromemco, Inc., 280 Bernardo Ave., Mountain View, CA 94043, (415) 964-7400.

CIRCLE INQUIRY NO. 139

P.O. Box 4430N Santa Clara, CA 95054
For will call only: **(408) 988-1640**
2322 Walsh Ave.

Quest ELECTRONICS



RCA Cosmac Super Elf Computer $106.95

Compare features before you decide to buy any other computer. There is no other computer on the market today that has all the desirable benefits of the **Super Elf** for so little money. The Super Elf is a small single board computer that does many **big** things. It is an excellent computer for training and for learning programming with its machine language and yet it is easily **expanded** with **additional memory, Tiny Basic, ASCII Keyboards, video character generation, etc.**

The **Super Elf** includes a **ROM** monitor for program loading, editing and execution with **SINGLE STEP for program debugging** which is not included in others at the same price. With **SINGLE STEP** you can see the microprocessor chip operating with the **unique Quest** address and data bus displays **before, during** and **after** executing instructions. Also, CPU mode and instruction cycle are decoded and displayed on eight LED indicator lamps.

An RCA 1861 **video graphics** chip allows you to connect to your own TV with an inexpensive video modulator to do graphics and games. There is a **speaker system** included for writing your own music or using many music programs already written. The speaker amplifier may also be used to drive relays for control purposes.

A 24 key **HEX keyboard** includes 16 HEX keys plus **load, reset, run, wait, input, memory protect, monitor select** and **single step**. Large, on board displays provide output and optional **high and low address**. There is a 44 pin standard connector for PC cards and a 50 pin connector for the **Quest Super Expansion Board.** Power supply and sockets for **all** IC's are included in the price plus a detailed 127 pg. instruction manual which now includes over 40 pgs. of software info. including a series of lessons to help get you started and a music program and graphics target game.

Remember, other computers only offer Super Elf features at additional cost or not at all. **Compare before you buy. Super Elf Kit $106.95, High address option $8.95, Low address option $9.95. Custom Cabinet** with drilled and labelled plexiglass front panel **$24.95. NiCad Battery Memory Saver Kit $6.95.** All kits and options also come completely assembled and tested.

Questdata, a 12 page monthly software publication for 1802 computer users is available by subscription for $12.00 per year.

Attention Elf Owners

New products in hardware and software coming soon.

Tiny Basic cassette $10.00, on ROM $38.00, original Elf kit board $14.95.

Super Expansion Board with Cassette Interface $89.95

This is truly an astounding value! This board has been designed to allow **you** to decide how you want it optioned. **The Super Expansion Board comes with 4K of low power RAM** fully addressable anywhere in 64K with built-in memory protect and a **cassette interface**. Provisions have been made for all other options on the same board and it fits neatly into the hardwood cabinet alongside the **Super Elf**. The board includes slots for up to 6K of **EPROM** (2708, 2758, 2716 or TI 2716) and is **fully socketed.** EPROM can be used for the monitor and Tiny Basic or other purposes.

A IK Super ROM Monitor $19.95 is available as an on board option in 2708 EPROM which has been preprogrammed with a program loader/editor and error checking multi file cassette read/write software, (relocatable cassette file) another exclusive from Quest. It includes register save and readout, block move capability and video graphics driver with blinking cursor. Break points can be used with the register save feature to isolate program bugs quickly, then follow with single step. The Super Monitor is written with subroutines allowing users to take advantage of monitor functions simply by calling them up.

Improvements and revisions are easily done with the monitor. If you have the **Super Expansion Board** and **Super Monitor** the monitor is up and running at the push of a button.

Other on board options include **Parallel Input and Output Ports** with **full handshake**. They allow easy connection of an ASCII keyboard to the input port. **RS 232 and 20 ma Current Loop** for teletype or other device are on board and if you need more memory there are two **S-100** slots for static RAM or video boards. A Godbout 8K RAM board is available for $135.00. Also a 1K Super Monitor version 2 with video driver for full capability display with Tiny Basic and a video interface board. **Parallel I/O Ports $9.85, RS 232 $4.50, TTY 20 ma I/F $1.95, S-100 $4.50. A 50 pin connector set** with ribbon cable is available at $12.50 for easy connection between the **Super Elf** and the **Super Expansion Board.**

The **Power Supply Kit** for the Super Expansion Board is a 5 amp supply with multiple positive and negative voltages $29.95. Add $4.00 for shipping. Prepunched frame $5.00. Case $10.00. Add $1.50 for shipping.

Same day shipment. First line parts only. Factory tested. Guaranteed money back. Quality IC's and other components at factory prices.

INTEGRATED CIRCUITS

7400TTL	
7400N	.17
7402N	.17
7404N	.19
7409N	.23
7410N	.17
7414N	.63
7420N	.17
7422N	1.39
7430N	.20
7442N	.50
7445N	.69
7447N	.60
7448N	.69
7450N	.17
7474N	.29
7475N	.49
7485N	.88
7489N	2.00
7490N	.43
7492N	.43
7493N	.43
7495N	.69
74100N	.90
74107N	.29
74121N	.34
74123N	.59
74125N	.39
74145N	.69
74150N	.95
74151N	.69
74154N	1.00
74157N	.69
74161N	.87
74162N	.87
74163N	.87
74174N	.96
74175N	.90
74190N	1.15
74192N	.87
74193N	.85
74221N	1.55
74298N	1.65
74365N	.95
74366N	.95
74367N	.95

74LS00 TTL	
74LS00N	.25
74LS02N	.25
74LS04N	.25
74LS05N	.25
74LS08N	.25
74LS10N	.25
74LS13N	.40
74LS14N	.90
74LS20N	.25
74LS22N	.25
74LS28N	.41
74LS30N	.25
74LS33N	.39
74LS38N	.30
74LS74N	1.15
74LS75N	.47
74LS90N	.51
74LS93N	.51
74LS95N	1.89
74LS107N	.35
74LS112N	.35
74LS113N	.35
74LS132N	.72
74LS136N	.35
74LS151N	.67
74LS155N	.67
74LS157N	.67
74LS162N	.91
74LS163N	.91
74LS174N	.95
74LS190N	1.06
74LS221N	1.95
74LS258N	.67
74LS367N	1.35

LINEAR	
CA3045	.90
CA3046	.67
CA3081	1.80
CA3082	1.90
CA3089	2.95
LM301AN/AH	.35
LM305H	.87
LM307N	.35
LM308N	.89
LM309H	1.15
LM309K	1.50
LM311H/N	.90
LM317T/K	2.92
LM318	1.35
LM320K-5	1.60
LM323K-5	6.95
LM320K-12	1.60
LM320K-15	1.60
LM320T-5	1.60
LM320T-8	1.60
LM320T-12	1.50
LM320T-15	1.60
LM324N	1.15
LM339N	1.55
LM340K-5	1.35
LM340K-8	1.35
LM340K-12	1.35
LM340K-15	1.35
LM340K-24	1.35
LM340T-5	1.25
LM340T-8	1.25
LM340T-12	1.25
LM340T-15	1.25
LM340T-18	1.25
LM340T-24	1.25
LM343H	4.50
LM350	7.50
LM370	1.15
LM377	3.00
LM379M	5.00
LM380N	1.00
LM381	1.60
LM382	1.60
LM703H	.40
LM709H	.28
LM723H/N	.50
LM733N	.67
LM741CH	.35
LM741N	.25
LM747H/N	.62
LM748N	.35
LM1303N	.82
LM1304	1.10
LM1305	1.27
LM1307	2.00
LM1310	2.75
LM1458	.47
LM1800	1.75
LM1812	7.50
LM1889	3.00
LM2111	1.75
LM2902	1.50
LM3900N	.60
LM3905	1.75
LM3909N	.61
MC1458V	.50
NE550N	.65
NE555V	.43
NE556A	.79
NE565A	1.00
NE566V	1.50
NE567V	1.20
NE570B	5.00
NE571B	5.00
78L05	.60
78L08	.60
79L05	.70
78M05	.85
75108	1.75
75491CN	.50
75492CN	.55
75494CN	.89

A to D CONVERTER	
8038B	4.50
8700CJ	13.95
8701CN	15.95
8750CJ	13.95
LD130	9.95
9400CJV/F	7.40
ICL7103	9.50
ICL7107	14.25
8702	17.95

CMOS	
CD34001 Fair.	.50
CD4000	.16
CD4001	.21
CD4002	.21
CD4006	1.10
CD4007	.21
CD4008	.21
CD4009	.39
CD4010	.39
CD4011	.21
CD4012	.21
CD4013	.36
CD4014	.86
CD4015	.86
CD4016	.36
CD4017	.94
CD4018	.94
CD4019	.21
CD4020	1.02
CD4021	1.02
CD4022	.86
CD4023	.21
CD4024	.75
CD4025	.21
CD4026	1.51
CD4027	.36
CD4028	.79
CD4029	1.02
CD4030	.21
CD4035	1.02
CD4040	1.02
CD4042	.71
CD4043	.63
CD4044	.63
CD4046	1.67
CD4049	.36
CD4050	.36
CD4051	1.13
CD4060	1.42
CD4066	.71
CD4068	.40
CD4069	.40
CD4070	.40
CD4071	.21
CD4072	.21
CD4073	.21
CD4075	.21
CD4076	1.75
CD4078	.40
CD4081	.21
CD4082	.21
CD4116	.47
CD4490	5.50
CD4507	1.00
CD4508	4.25
CD4510	1.02
CD4511	.94
CD4515	2.52
CD4516	1.10
CD4518	1.02
CD4520	1.02
CD4527	1.51
CD4528	.79
CD4553	3.50
CD4566	2.25
CD4583	4.50
CD4585	1.10
CD40192	3.00
74C00	.28
74C04	.33
74C10	.28
74C14	2.10
74C20	.28
74C30	.28
74C48	1.95
74C74	.75
74C76	1.40
74C90	1.15
74C93	1.40
74C154	3.00
74C160	1.44
74C175	1.35
74C192	1.65
74C221	2.00
74C905	3.00
74C906	.75
74C914	1.95
74C922	5.50
74C923	5.50
74C925	6.95
74C926	6.95
74C927	6.95

INTERFACE	
8095	.95
8096	.95
8097	.95
8098	.95
8T09	1.25
8T10	4.50
8T13	3.00
8T20	5.50
8T23	3.10
8T24	3.50
8T25	3.20
8T26	1.69
8T28	2.75
8T97	1.69
8T98	1.69

MOS/MEMORY RAM	
2101-1	3.95
2102-1	.95
2102AL-4	1.60
21L02-1	1.18
21F02	1.25
2104A-4	4.95
2107B	4.95
2111-1	3.75
2112-2	3.95
2114L-3	6.75
4116	10.95
2513B	6.30
MM5262	.40
MM5280	3.00
MM5320	9.95
MM5330	5.94
PD411D-3	4.00
PD411D-4	5.00
P5101L	13.95
4200A	9.95
82S25	2.90
91L02A	1.50
HD0165-5	6.95
MM57100	4.50
GIAY38500-1	9.95
MCM6571A	9.95
9368	3.50
4100	10.00
416	16.00

CLOCKS	
MM5309	3.00
MM5311	3.60
MM5312	4.80
MM5313	3.60
MM5314	3.90
MM5315	4.00
MM5316	3.60
MM5318	3.60
MM5369	2.10
MM5841	14.45
MM5865	7.95
CT7001	5.80
CT7002	10.95
CT7010	8.95
CT7015	8.95
MM5375AA/N	3.90
MM5375AB/N	4.90
7205	16.50
7207	7.50
7208	15.95
7209	4.95
DS0026CN	3.75
DS0056CN	3.75
MM53104	2.50

MICROPROCESSOR	
6800	17.50
6802	24.95
8080A with data	8.95
8085	27.00
Z80A	19.75
8212	2.90
8214	8.00
8216	2.90
8224	2.90
8228	5.35
8251	8.50
8253	10.00
8255	9.25
8257	19.50
8259	19.50
1802CP plas.	13.95
1802DP plas.	17.95
1861P	11.50
CDP1802CD	19.95
CDP1802D	25.00
CDP1861	12.95
6820	9.95
6850	12.95
6502	12.50

IC SOCKETS
Solder Tin Low Profile

PIN	1UP	PIN	1UP
8	.15	22	.30
14	.14	24	.35
16	.16	28	.42
18	.27	36	.58
20	.29	40	.57

3 level wire wrap gold.
14 pin .25
2 level 14 pin ww .20

WIRE WRAP LEVEL 3

PIN		PIN	
14	.25	24	.86
16	.33	28	1.00
18	.57	40	1.23

UART/FIFO	
AY5-1013	5.50
AY5-1014	7.50
3341	6.95

PROM	
1702A	3.95
N82S23	2.95
N82S123	3.50
N82S126	3.75
N82S129	3.75
N82S131	3.75
N82S136	8.75
N82S137	8.75
2708	10.50
DM8577	2.90
8223	2.90
2716T1	29.50
2716 Intel	48.00

CRYSTALS	
1 MHz	4.50
2 MHz	4.50
4 MHz	4.25
5 MHz	4.25
10 MHz	4.25
18 MHz	3.90
20 MHz	3.90
32 MHz	3.90
32768 MHz	4.00
1.8432 MHz	4.50
3.5795 MHz	1.20
2.0100 MHz	1.95
2.097152 MHz	4.50
2.4576 MHz	4.50
3.2768 MHz	4.50
5.0688 MHz	4.50
5.185 MHz	4.50
5.7143 MHz	4.50
6.5536 MHz	4.50
14.31818 MHz	4.25
18.432 MHz	4.50
22.1184 MHz	4.50

CONNECTORS	
44 pin edge	2.75
100 pin edge	4.50
100 pin edge WW	4.75

KEYBOARD ENCODERS	
AY5-2376	$12.50
AY5-3600	13.50
74C922	5.50
74C923	5.50
HD0165-5	6.95

IC Test Clips

	1	10
Red	.55	.47
Black	.55	.47

Keyer 8043 14.50
comp. w/spec./sock.

TRANSISTORS	
2N1893	.40
2N2222A	.18
2N2369	.30
2N2904A	.20
2N2907A	.25
2N3053	.40
2N3638	.25
2N3643	.25
2N3904	.18
2N3906	.18
2N3055	.69
2N4400	.25
2N4401	.75
2N4402	.20
TIP31	.60
TIP33A	1.00

D Connectors RS232
25 Pin Subminiatures

DB25P	2.95
DB25S	3.95
Cover	1.50
RS232 Complete Set	6.50
DE9P	1.50
DE9S	1.95
DA15P	2.10
DA15S	3.10

CLOCK MODULES Complete alarm clocks ready to hook up with transformer and switches. Very compact with .50" and .84" digits.

MA1002A, C or E .50"	**8.95**
102P3 Transformer	**2.25**
MA1010A, C or E .84"	**11.95**
102P2 Transformer	**2.25**
Special transformer and six switches when purchased w/module	**2.95**
MA1003 car module .3" green fluor. display	**15.95**

RESISTORS ¼ watt 5%
10 per type .03 — 1000 per type .012
25 per type .025 — 350 piece pack
100 per type .015 — 5 per type 6.75

KEYBOARDS	
56 key ASCII keyboard kit	$67.50
Fully assembled	77.50
53 key ASCII keyboard kit	60.00
Fully assembled 70.00 Enclosure	14.95

LEDS	
Red T018	.15
Green, Yellow T018	.20
Jumbo Red	.20
Green, Orange, Yellow Jumbo	.25
Cliplite LED Mounting Clips	8/$1.25

(specify red, amber, green, yellow, clear)

CONTINENTAL SPECIALTIES in stock
Complete line of breadboard test equip.
MAX-100 8 digit Freq. Ctr. $128.95

OK WIRE WRAP TOOLS in stock
Portable Multimeter $18.00

DIGITAL THERMOMETER $48.50
Batt. oper. General purpose or medical 32°–230°F. Disposable probe cover ±.2° accuracy. Comp. Assy. in compact case. Switches from F° to C°.

COMPUTER BOARD KITS	
8K RAM Board Kit	$135.00
4K EPROM Kit	114.95
I/O Board Kit	44.50
Extender Board w/connector	12.50
16K EPROM board kit w/o PROMS	74.50
North Star Floppy Disk Kit	$665.00
Additional Drive Kit	415.00

SPECIAL PRODUCTS	
MM5865 Stopwatch Timer	9.00
PC board	7.50
Switches Mom. Pushbutton	.27
3 pos. slide	.25
Encoder HD0165-5	6.95
3 Digit Universal Counter Board Kit Operates 5–18 Volt DC to 5 MHz typ. .125" LED display	**10.50**
Voice actuated switch	.50
Paratronics 100A Logic Analyzer Kit	$235.00
Model 10 Trigger Expander Kit	$229.00
Model 150 Bus Grabber Kit	$369.00
Sinclair 3½ Digit Multimeter	**$59.95**
Clock Calendar Kit	**$23.95**

TRANSFORMERS	
6V 300 ma	3.25
12 Volt 300 ma transformer	1.25
12.6V CT 600 ma	3.75
12V 250 ma wall plug	2.95
12V CT 250 ma wall plug	3.50
24V CT 400 ma	3.95
10V 1.2 amp wall plug	4.85
12V 6 amp	12.95

DISPLAY LEDS			
MAN1	CA	.270	2.90
MAN3	CC	.125	.39
MAN72/74	CA/CA	.300	1.00
DL704	CC	.300	1.25
DL707/DL707R	CA	.300	1.00
DL727/728	CA/CC	.500	1.90
DL747/750	CA/CC	.600	1.95
DL750	CC	.600	1.95
FND359	CC	.357	.70
FND500/507	CC/CA	.500	1.35
FND503/510	CC/CA	.500	.90
FND800/807	CC/CA	.800	2.20
3 digit Bubble			.60
4 digit bubble			.80
DG8 Fluorescent			1.75
DG10 Fluorescent			1.75
5 digit 14 pin display			1.00
NSN69 9 digit display			.60
7520 Clairex photocells			.39
TIL311 Hex			9.50

COMPUTER GRADE CAPS			
1600	mfd	200V	4.50
2000	mfd	45V	2.50
3200		50V	2.50
5500		25V	2.50
5800		40V	3.00
6100		40V	3.00
6100		50V	3.00
7700		50V	2.50
8000		55V	2.50
10000		20V	2.50
12000		30V	3.50
35000		25V	5.00
55000		25V	5.00
82000		15V	6.00
60000		40V	6.00
76000		40V	6.00
120000		10V	6.00

Auto Clock Kit $15.95
DC clock with 4-.50" displays. Uses National MA-1012 module with alarm option. Includes light dimmer, crystal timebase PC boards. Fully regulated, comp. instructs. Add $3.95 for beautiful dark gray case. Best value anywhere.

RCA Cosmac VIP Kit $229.00
Video computer with games and graphics. Fully assem. and test. **$249.00**

Not a Cheap Clock Kit $14.95
Includes everything except case. 2-PC boards. 6-.50" LED Displays. 5314 clock chip, transformer, all components and full instructions. Orange displays also avail. **Same kit w/.80" displays.** Red only. **$21.95** Case **$11.75**

60 Hz Crystal Time Base Kit $4.40
Converts digital clocks from AC line frequency to crystal time base. Outstanding accuracy. Kit includes: PC board, IC, crystal, resistors, capacitors and trimmer.

Digital Temperature Meter Kit
Indoor and outdoor. Switches back and forth. Beautiful. 50" LED readouts. Nothing like it available. Needs no additional parts for complete, full operation. Will measure −100° to +200°F, tenths of a degree, air or liquid. Very accurate. **$39.95**
Beautiful woodgrain case w/bezel **$11.75**

NiCad Battery Fixer/Charger Kit
Opens shorted cells that won't hold a charge and then charges them up, all in one kit w/full parts and instructions. **$7.25**

PROM Eraser
Will erase 25 PROMs in 15 minutes. Ultraviolet, assembled **$34.50**

Rockwell AIM 65 Computer
6502 based single board with full ASCII keyboard and 20 column thermal printer. 20 char. alphanumeric display, ROM monitor, fully expandable. $375.00. 4K version $450.00. 4K Assembler $85.00, 8K Basic Interpreter $100.00. Power supply assembled in case $60.00.

Multi-volt Computer Power Supply
8v 5 amp, ±18v .5 amp, 5v 1.5 amp, −5v .5 amp, 12v .5 amp, −12 option. ±5v, +12v are regulated. Kit $29.95. Kit with punched frame $34.95. Woodgrain case $10.00.

Video Modulator Kit $8.95
Convert your TV set into a high quality monitor without affecting normal usage. Complete kit with full instructions.

2.5 MHz Frequency Counter Kit
Complete kit less case **$37.50**

30 MHz Frequency Counter Kit
Complete kit less case **$47.75**
Prescaler kit to 350 MHz **$19.95**

79 IC Update Master Manual $35.00
Complete IC data selector, 2500 pg. master reference guide. Over 50,000 cross references. Free update service through 1979. Domestic postage $3.50. Foreign $5.00. 1978 IC Master closeout $19.50.

Stopwatch Kit $26.95
Full six digit battery operated. 2–5 volts. 3.2768 MHz crystal accuracy. Times to 59 min., 59 sec., 99 1/100 sec. Times std., split and Taylor. 7205 chip, all components minus case. Full instructions.

Hickok 3½ Digit LCD Multimeter
Batt/AC oper. 0.1mv–1000v. 5 ranges. 0.5% accur. Resistance 6 low power ranges 0.1 ohm–20M ohm. DC curr. .01 to 100ma. Hand held, ½" LCD displays, auto zero, polarity, overrange. $74.95.

S-100 Computer Boards

8K Static RAM Kit Godbout	$135.00
16K Static RAM Kit	265.00
24K Static RAM Kit	423.00
32K Dynamic RAM Kit	310.00
64K Dynamic RAM Kit	470.00
8K/16K Eprom Kit (less PROMS)	**$89.00**
Video Interface Kit	**$139.00**

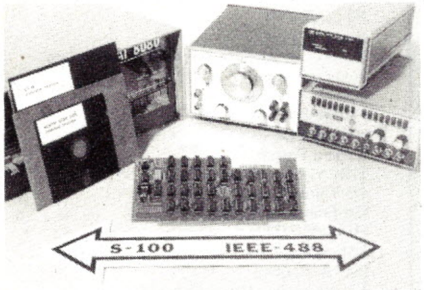
Motherboard **$39.** Extender Board **$8.99**

TERMS: $5.00 min. order U.S. Funds. Calif residents add 6% tax.
BankAmericard and Master Charge accepted.
Shipping charges will be added on charge cards.

FREE: Send for your copy of our NEW 1979 QUEST CATALOG. Include 28¢ stamp.

I/O Board

IEEE-488 instruments can now be controlled from high level languages (like BASIC) using inexpensive S-100 hardware. New software support packages for the P&T-488 interface board allow programs that use the normal I/O structure of the computer system to function on the 488 bus as a talker, listener, or controller.

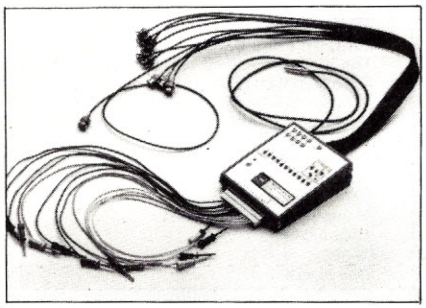


Price of the P&T-488 interface board is $400 assembled and tested with one software package. For details contact Pickles & Trout, P.O. Box 1206, Goleta, CA 93017, (805) 967-9563.

CIRCLE INQUIRY NO. 140

New E-H Data Probe

The Model DP11 is an optional data probe to provide sequential and expanded combinational triggering capability for E-H International's LA1850 logic analyzer.



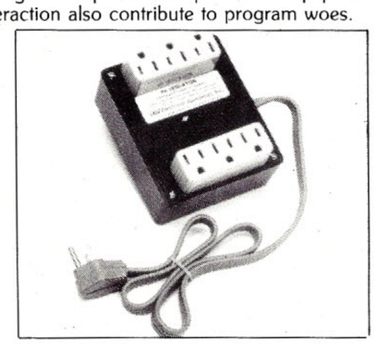
The DP11 has eleven input channels, eight of which pass through to the logic analyzer. The other three inputs may be used as signal qualifiers. A clock input is also included which may pass straight through or be qualified.

Price is 525. For details contact E-H International, Inc., 515 11th St., Oakland, CA 94607.

CIRCLE INQUIRY NO. 141

Don't Blame the Software

Crashes, memory loss and program glitches are often caused by AC power line hash and surges. Peripheral and processor equipment interaction also contribute to program woes.



To combat these problems, Electronic Specialists has expanded its ISOLATOR line. Model ISO-2 comprises two filtered banks of 3-prong AC sockets (6 sockets total) with integral surge suppression.

Connecting to the 125 VAC power line with a standard 3-prong plug, Model ISO-2 can isolate and protect an 1875 watt total load, with each socket bank capable of isolating a 1000 watt load.

Price is $49.95. For details contact Electronic Specialists, Inc., Box 122, Natick, MA 01760.

CIRCLE INQUIRY NO. 142

Language for Personal Computers

SLIC, a high-level structured language for personal computers, combines the simplicity of BASIC with the clarity of expression of structured languages like C and Pascal. Unlike compiler languages, SLIC is designed for interactive program development. SLIC also provides an ideal way to learn the modern techniques of top-down design and structured programming.

Among SLIC's features are: GOTO statements eliminated for clearer code, three types of loops (while, repeat-until, for), generalized if-else with statement grouping, functions with arguments (similar to FORTRAN subroutines), unlimited length character strings, and extended-precision math functions.

TAPE SLIC is available in either TRS-80 or Tarbell cassette formats for $50. DISK SLIC is available for CP/M users at $95. Prices include a self-teaching user's manual, which may be ordered separately for $10. For more information contact RTG Data Systems, 1003 Wilshire Blvd., Suite 202, Santa Monica, CA 90401.

CIRCLE INQUIRY NO. 152

Data Base Management Software for 6800 Computers

FINDER™ (File INformation on Disk for Easy Retrieval) is a general purpose data base management program for 6800 microcomputers using PerCom's LFD-400 mini-floppy disk systems.

The program allows users to define and access the items of databases using their own terminology, and to customize file structures to their particular requirements.

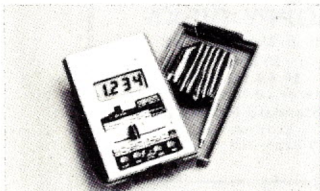
Written in PerCom Super BASIC, Finder requires 24K bytes of memory. Price is $99.95 on mini-diskette and users manual. For details contact PerCom Data Co., 318 Barnes, Garland, TX 75042, (214) 272-3421. To order by phone call 1-800-527-1592.

CIRCLE INQUIRY NO. 154

LX303

ALL THE MOST WANTED FEATURES IN A COMPACT DVOM

BIG ½" HIGH LCD DISPLAY
USE INDOORS OR OUT
200 HOUR 9V BATTERY LIFE
AUTO ZERO, POLARITY, OVERRANGE INDICATION
100 mV DC F.S. SENSITIVITY
19 RANGES AND FUNCTIONS



Removable cover stores test leads set furnished as part of the unit.

Available accessories include AC adapter, padded vinyl carrying case, 40KV DC probe, 10 Amp DC shunt.

X10 DCV probe adapter available for protecting input up to 10KV.

ONLY $74.95

HICKOK

THE HICKOK ELECTRICAL INSTRUMENT CO.
10514 Dupont Avenue • Cleveland, Ohio 44108
(216) 541-8060 • TWX: 810-421-8286

On-the-Spot accuracy, wherever and whenever you need it. The Hickok LX303 is ideal for any field service, industrial maintenance or personal application. Rugged, Reliable. Easy to read in any light, this exciting, new, 3½ digit Mini-Multimeter weighs only 12 ounces and carries a full one year guarantee. Features previously found only in expensive units . . . at a price under $75.00! Another American made test equipment breakthrough from Hickok, The Value Innovator for over 60 years. Order Today!

ickok distributor or order below

SPECIFICATIONS:
DC VOLTS (5 RANGES): 0.1mV to 1000V; Accuracy ±0.5% rdg ±0.5% f.s.; Input imped: 10MΩ; Max. input 1kV except 500V on 200mV range.
AC VOLTS (40Hz to 5kHz): 0.1V to 600V; Accuracy: ±1.0% rdg ±0.5% f.s. (−2dB max. at 5kHz); Max. input: 600V.
RESISTANCE (6 LOW POWER RANGES): 0.1Ω to 20MΩ; Accuracy: ±0.5% rdg ±0.5% f.s. (±1.5% rdg on 20MΩ range); input protected to 120VAC all ranges.
DC CURRENT (6 RANGES): .01nA to 100mA; Accuracy: ±1.0% rdg ±0.5% f.s.
DIMENSIONS AND WEIGHT: 5-7/8" x 3-3/8" x 1-3/4", 12 oz.; POWER: 9V batt. (not incl.) or Hickok AC adapter; READ RATE: 3/sec. OPERATING TEMPERATURE: 0°-50°C.

JADE Computer Products
800/421-5809 Continental U.S.
800/262-1710 Inside California
4901 W. ROSECRANS AVENUE
Department "B" 3
HAWTHORNE, CALIFORNIA 90250

PLEASE SEND ME
_______ Hickok LX303 Digital Multimeters .... @ 74.95 ea.
_______ RC-3 AC Adapter, 115VAC (220VAC avail.) .... @ 7.50 ea.
_______ CC-3 Deluxe Carrying Case .... @ 7.50 ea.
_______ VP-10 X10 DCV Probe Adapter .... @ 14.95 ea.
_______ CS-1 10A DC Current Shunt .... @ 14.95 ea.
_______ VP-40 40 KV DC Probe .... @ 35.00 ea.

☐ Bill my company, P. O. attached (D & B rated firms only) net 30 days
☐ Payment enclosed Bill my: ☐ Master Charge ☐ VISA

Account No. ____________ Exp. Date ________
Name ____________ Signature ____________
Address ____________
City ____________ State ________ Zip ________
Add $3.00 Postage and Handling. **CALIFORNIA RESIDENTS ADD 6% SALES TAX**

PLACE ORDERS TOLL FREE: 800/421-5809 Continental U.S. 800/262-1710 Inside California

LEEDEX MONITOR

- 12" Black and White
- 12MHZ Bandwidth
- Handsome Plastic Case

$139.00

THE KIM 1

KIM-1 Module monitor, programs stored in 2048 ROM Bytes, User Manual, Wall size schematic, Hardware manual, Programming manual, Programmers reference card, and Keyboard display.

$179

CASES — $29.95

THE SYM-1 $245

6502 — based single board computer with keyboard/display, KIM-1 hardware compatible, complete documentation.

SYM—1 CASE $39.95

MICROPROCESSORS	
F8	$16.95
Z80 (2MHz)	$20.00
Z80A (4MHz)	$14.95
CDP1802CD	$17.95
6502	$11.95
6800	$16,95
6802	$25.00
8008-1	$12.95
8035	$20.00
8035-8	$21.00
8080A	$10.00
8085	$23.00
TMS9900TL	$49.95
8080A SUPPORT DEVICES	
8212	$ 2.90
8214	$ 4.65
8216	$ 2.75
8224 (2MHz)	$ 4.30
8224-4 (4MHz)	$ 9.95
8226	$ 2.75
8228	$ 6.40
8238	$ 6.40
8243	$ 8.00
8251	$ 7.50
8253	$20.00
8255	$ 6.45
8257	$20.00
8259	$20.00
8275	$75.00
8279	$18.50
USRT	
S2350	$10.95
UARTS	
AY5-1013A	$ 5.25
AY5-1014A	$ 8.25
TR1602B	$ 5.25
TMS6011	$ 5.95
IM6403	$ 9.00
BAUD RATE GENERATORS	
MC14411	$10.00
14411 Crystal	$ 4.95
6800 PRODUCT	
6810P	$ 4.00
6820P	$ 6.60
6821P	$ 6.60
6828P	$11.25
6834P	$16.95
6850P	$ 8.65
6852P	$11.00
6860P	$ 9.25
6862P	$12.00
6871P	$28.75
6875P	$ 8.75
6880P	$ 2.50
CHARACTER GENERATORS	
2513 Upper (-12+5)	$ 6.75
2513 Lower (-12+5)	$ 6.75
2513 Upper (5 volt)	$ 9.75
2513 Lower (5 volt)	$10.95
MCM6571 up scan	$10.95
MCM6571A down scn	$10.95
PROMS	
1702A	$5.00
2708	$ 9.95
2716 (5+12) TI	$60.00
2716 (5v) INTEL	$60.00
2758 (5v)	$30.00
DYNAMIC RAMS	
416D/4116 (200ns)	$12.50
2104/4096	$ 4.00
2107B-4	$ 3.95
TMS4027/4096 (300ns)	$ 4.00

STATIC RAMS	1—15	16—100
21L02 (450ns)	$ 1.50	$ 1.20
21L02 (250ns)	$ 1.75	$ 1.50
2101-1	$ 2.95	$ 2.60
2111-1	$ 3.25	$ 3.00
2112-1	$ 2.95	$ 2.65

FLOPPY DISK CONTROLLERS	
1771B01	$39.95
1791	$49.95
KEYBOARD CHIPS	
AY5-2376	$13.75
AY5-3600	$13.75
MM5740	$18.00

JADE'S *DOUBLE DENSITY Controller Board*

KIT: $249.00
Assmb. & Tstd: $299.00

- Single or Double Density Recording
- Full Size or Mini Floppy
- CP/M Compatable in either density
- Programmed Data Transfer, no DMA
- Controls up to 8 drives
- IBM format in either density
- Software Selectable Density

■ This controler utilizes the proven reliability of the IBM standard format as well as the lastest phase-locked-loop for data seperation. ■ All clocks are generated from an on-board cyrstal oscillator ■ Right precompensation is used to enhanced data recovery reliability in the double density mode ■ Density selection is enitirely transparent to the user ■ Single and double density diskettes can be mixed on the same system.

NEW! JADE Z80 BOARD

IMPROVED DESIGN AND FEATURES

- ON BOARD 2708 or 2716 EPROM
- VERY RELIABLE AT 4 MHZ OR 2 MHZ
- POWER ON JUMP AND RESET
- ON BOARD USART (8251)

2MHz	
Kit	$135.00
Assembled & Tested	$185.00
4 MHz	
Kit	$149.95
Assembled & Tested	$199.95
Bare Board	$ 35.00

JADE 8080A

with full documentation

Kit	$100.00
Assembled & Tested	$149.95
Bare Board	$ 30.00

EPROM BOARD KITS

JG8/16 (uses 2708 or 2716) **$69.95**

APPLE•TRS-80•APPLE•TRS-80•APPLE•TRS-80

Jade Memory Expansion Kits for TRS-80 and Apple!

4116's

Everything a person needs to add 16K of memory. Chips come neatly packaged with easy to follow directions. In minutes your machine is ready for games and more advanced software.

$82.00

"IMSAI"-TYPE CARD GUIDE SPECIAL:

Regular Price 30¢ each

SPECIAL: 10 for $1.00!

Rockwell AIM-65: The Head-Start in Microcomputers

A KIM-1 compatible machine with on-board printer and a real keyboard!

$375.00 w/1K RAM
$450.00 w/4K RAM

4K assembler/editor in ROM:	$ 80.00
8K BASIC in ROM:	$100.00
Power supply:	$ 59.95
Case for AIM-65:	$ 49.95

Special Package Price: $599.00
AIM-65 (4K), Power Supply, Case, and 8K BASIC ROM

New Prices

DYNAMIC RAM BOARDS EXPANDABLE TO 64K

32K VERSION • KITS

Uses 4115 (8Kx1, 250ns) Dynamic RAM's, can be expanded in 8K increments up to 32K:

8K	$159.00	24K	$249.00
16K	$199.00	32K	$299.00

4115 SALE 8 for $39.95

64K VERSION • KITS

Uses 4116 (16Kx1, 200ns) Dynamic RAM's, can be expanded in 16K increments up to 64K:

16K	$249.00	48K	$469.00
32K	$369.00	64K	$569.00

DYNABYTE *Naked Terminal*

$350.00

FEATURES & BENEFITS
■ 80 character by 24 line format (Model 57) ■ Completely self contained terminal electronics, just add CCTV monitor and key board ■ No support software required ■ Switch selectable modes: Half Duplex, Full Duplex, Block mode ■ Block mode allows for editing before transmit ■ Keyboard interface provided ■ Fully assembled, socketed, tested, burned in, and guaranteed for a full year from date of purchase ■ Video is switch selectable "Black-on-White" or "White-on-Black"

TEXTOOL ZIP* DIP® II Sockets

***ZERO INSERTION FORCE sockets**

PRICES:

16 pin Zip Dip II	$5.50
24 pin Zip Dip II	$7.50
40 pin Zip Dip II	$10.25

S-100 EDGE CONNECTORS

Soldertail $3.25 each
10 for $35.00
Wire Wrap $4.00 each
10 for $40.00

★ STATIC RAM SPECIALS ★

2114's, low power (1024x4)	1-15	16-99	100 +
450ns	8.00	6.95	5.50
300ns	9.00	8.00	6.50
TMS4044/MM5257, low power			
450ns	8.00	7.50	6.50
300ns	9.95	8.75	8.00
4200A (4Kx1, 200ns)	9.95	8.50	8.00
410D (4K x 1, 200 ns)	8.25	7.00	6.75

STATIC RAM BOARDS

JADE 8K

Kits:	450ns	$125.95
	250ns	$149.75
Assembled & Tested:		
	450ns	$139.75
	250ns	$169.75
Bare Board:		$ 25.00

16K — Uses 2114's (low power)
Assembled & Tested:

RAM 16 (250ns)	$375.00
RAM 16B (450ns)	$325.00

16K with memory management
Assembled & Tested:

RAM 65 (250ns)	$390.00
RAM 65B (450ns)	$350.00

32K Static
Assembled & Tested:

250ns	$795.00
450ns	$725.00
250ns Kit	$575.00

JADE Computer Products

4901 W. ROSECRANS AVENUE
Department "F" 3
HAWTHORNE, CALIFORNIA 90250
U.S.A.

Telephone:
(213) 679-3313
(800) 421-5809 Continental U.S.
(800) 262-1710 Inside California

Cash, checks, money orders, and credit cards accepted. Add freight charge of $2.50 for orders under 10 lbs. and $1.00 service charge for orders under $10.00. Add 6% sales tax on all parts delivered in California. Discounts available at OEM quantities.

WRITE FOR OUR FREE CATALOG
All prices subject to change without notice.

Terminals and Printers

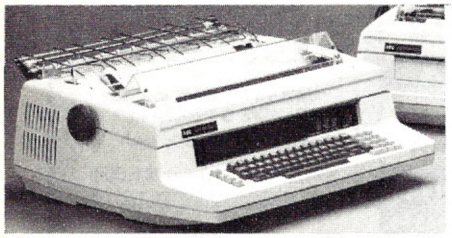
for every application . . .

NEC **spinwriter**™

RO's as low as **$1695**

KSR Terminal (55 cps, RS232C) as low as **$2595**

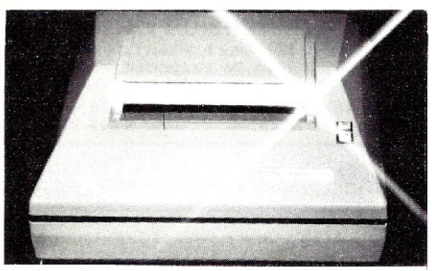
— Letter Quality at 55 cps —



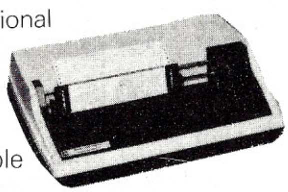
THE PRINTERM 879

Unbelievable speed at low price — 120 cps, printing in upper and lower case with a 9 x 7 dot matrix in a bi-directional mode. Microprocessor-controlled RS232 or optional parallel interface directly compatible with TRS-80 or Cromemco System 2 or 3. Friction or pin-feed models available. Friction-feed model can use standard Teletype roll.

Friction Feed Model, List $1395	$1099
Pin-Feed Model, List $1495	$1199



T.I. 810 as low as $1695



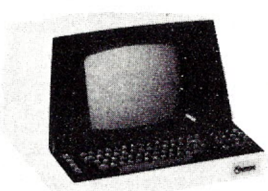
Bi-directional

165 cps

Logic-Seeking

Adjustable Tractor

SOROC 120



NEW LOW PRICE

$795

INTERTUBE II by Intertec

- 12" Display
- 24 x 80 format
- 18-key numeric keypad
- 128 upper/lower case ASCII characters
- Reverse video, blinking
- Complete cursor addressing and control
- Special user-defined control function keys
- Protected and unprotected fields
- Line insert/delete and character insert/delete editing
- Eleven special line drawing symbols

List $995

OUR PRICE **$784** Limited Time Only!

BANTAM 550

from PERKIN ELMER

Small in size, light in weight, and low in price — but on top of the list in features and performance.

- Upper and lower case
- Full 24 x 80 format
- Sharp 7 x 10 dot matrix

Get everything you want without paying for things you don't need — List: $996

NOW FROM US AT **$799**

Add $20 for anti-glare screen

Integral Data Systems, Inc.

IP-125 with 1210 Option*

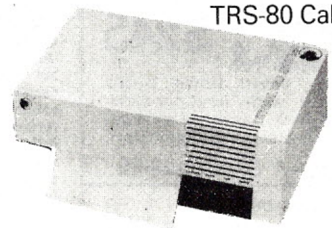
List $838 OUR PRICE **$754**

IP-225 with Tractor, 1210 and 1250 options

List $984 **$889**

*1210 Option is expanded and compressed print

TRS-80 Cable — $49

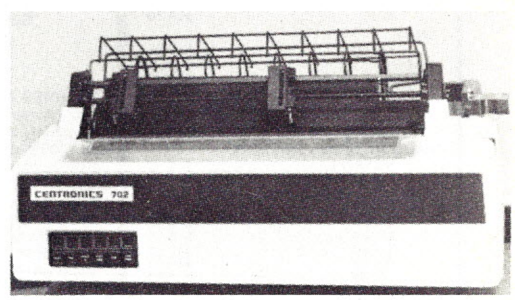


CENTRONICS

779 (60 cps — same as TRS-80 printer OUR PRICE ONLY **$979**

979 with Tractor **$1039**

702 (120 cps, bi-directional)	$1895
703 (185 cps, bi-directional)	$2189



OTHER VALUES . . .

DECwriter II $1490

Teletype 43 Printers as low as $880

Hazeltine 1500 our price $1049

Lear Siegler ADM3A $849

SHIPPING, HANDLING and INSURANCE: Intertube, Bantam 550, Printerm, IP-125/225, Teletype 43's, Hazeltine 1500, and ADM3A can be shipped by UPS. Heavier printers — air or truck, freight collect.

All prices subject to change and all offers subject to withdrawal without notice.

WRITE FOR FREE CATALOG

CIRCLE INQUIRY NO. 98

MiniMicroMart, Inc. 1618 James St., Syracuse, NY 13203 (315) 422-4467 TWX 710 541-0431

Home Poison Control

Use Your Computer to Save a Life

Review by Alan R. Miller, Software Editor

INTRODUCTION

You have just discovered that your child has swallowed a bottle of XYZ. What should be done? Should vomiting be induced? Unfortunately, this common situation results in more than 3,000 fatal poisonings every year in the United States alone. Most of the victims are less than five years old. If there is a computer in the home, the next step can be easier.

In contrast to the office computer which may be dedicated to business, the home computer is often used for playing games. One common game is called CHOMP in which the players take a byte out of a metaphorical cookie. The object is to avoid the poison. More serious programs for home computers have dealt with keeping financial records in order or helping to organize the meals.

A POISON-CONTROL PROGRAM

Berkeley Medical Data Associates, Inc., P.O. Box 5279, Berkeley, CA 94705, (415) 653-6707, offers a disk BASIC program called Home Poison Control. POISON is not a game, but rather a program that can provide help during an emergency. POISON is written in BASIC and is designed for direct and immediate interaction with a person who may have little knowledge of either medicine or computer operation.

The package contains a list of 422 entries, consisting of both generic and trade names of common household products. Most of the items are toxic substances, but there are also entries for relatively harmless things such as sand, dirt, tea, and yeast. Likely variants in spelling such as DRANO and DRAINO are provided for. The program does not require the user to evaluate the victim's symptoms, but only identify the poison involved.

POISON is available in two versions for the 8080 and Z-80 computers: a 5-inch, single-density North Star diskette, and an 8-inch, CP/M diskette for use with CBASIC (See the December 1978 issue of INTERFACE AGE.)

INITIALIZING THE NORTH STAR DISKETTE

To initialize the North Star version, the user must copy his own version of DOS and BASIC onto the first part of the POISON diskette. These routines have not been placed on the diskette because of copyright problems. Also, it is likely that North Star DOS would have to be altered for the user's particular system anyway.

Release 4 of North Star DOS and BASIC are required because POISON uses some of the advanced features in this package. Earlier versions can be upgraded to Release 4 by corresponding with North Star Computers.

AUTOMATIC STARTUP

One of the features of Release 4 is turnkey startup. The user can alter DOS and BASIC so that when the POISON diskette is in place, and the bootstrap at E900 HEX is executed, DOS will load BASIC, BASIC will load POISON, and POISON will automatically start up. Furthermore, if the CPU board has a jump-on-reset option, it is only necessary to turn on the computer, insert the POISON diskette, and push RUN. This important feature means that anyone in the household, even children, can easily utilize the program.

First make a working copy of the diskette, then put the original diskette away for a backup copy. If there is more than one drive, put the original diskette in Drive 1, and an initialized diskette in Drive 2. Give the command:

```
CD 1 2
```

to copy the entire diskette. If only one drive is available, copy as much as possible into memory with the read command:

```
RD 0 4000 100
```

Switch diskettes, then copy from memory to the new diskette:

```
WR 0 4000 100
```

with the write command. This pair of commands will have to be repeated until all 186 blocks have been transferred.

The next step is to copy BASIC and the personalized version of DOS to the new diskette. With two drives, place the original diskette in Drive 1 and the new diskette in Drive 2. Give the commands:

```
CF DOS  DOS,2
CF BASIC BASIC,2
```

With a single drive, the commands LF and SF are used:

```
LF BASIC 4000 (with original disk)
SF BASIC 4000 (with new disk)
LF DOS 4000 (with original disk)
SF DOS 4000 (with new disk)
```

Appendix I of the user's manual gives detailed instructions for altering DOS and BASIC so that the turnkey-startup feature can be implemented. Step 1, however, should explicitly remind the user that the working area used by BASIC must be enlarged before loading POISON:

```
MEMSET 35000
```

Also, step 3 could give the command for creating the new directory entry for POISON:

```
*CR POISON 88 14
```

so that it will start at address 14.

INITIALIZATION OF THE BASIC PROGRAM

POISON is initialized simply by starting it up (after enlarging the working area). A list of options is then presented on the system console:

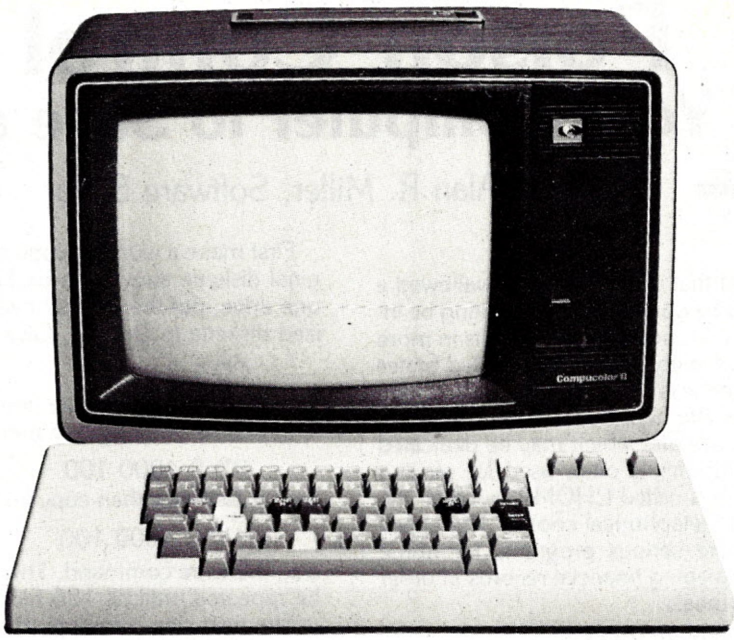
```
1. EMERGENCY TREATMENT
2. Change telephone numbers
3. List family allergies
4. Exit from program
   Type a number (1-4) ?
```

By entering a 2, the user can set the three telephone numbers:

```
Emergency Room number
Doctor's number
Poison Control number
```

Then by selecting option 3, a list of drug allergies for each family member can be entered.

Compucolor II



CRT Display
Eight color display with 32 lines of 64 characters (2048) characters). Two different character sizes. Plotting graphics of 128 x 128, including vector generating software. 64 standard ASCII characters and 64 additional special graphic chararters. Includes a Standard RS232C Terminal Mode for time sharing use. 60Hz refresh. Usable screen area 9" wide x 6⅜" high.

Microcomputer
Central Processing Unit: 8080A, 2 microsecond cycle time with total memory expandable to 64K bytes.

I/O
Input/Output Ports: system is designed for 478 ports, with 30 ports implemented in standard unit. Including one RS-232C Serial Asynchronous Channel for a printer or modem.
Baud Rate: Independent Baud rate generators for one of 7 Baud rates from 110 Baud to 9.6K Baud.

Keyboard
Separate keyboard with Standard ASCII 4 level, coded with 192 codes. Includes 71 gold crossbar commercial key switches. CPU Reset and Automatic disk loading (AUTO) keys are included. Optional: 101 keys with color and numeric clusters or 117 keys with 16 additional function keys.

Language
Language: DISK BASIC 8001 interpreter in ROM memory includes: 29 statement types: CLEAR, DATA, DEF, DIM, END, FILE, FOR, GET, GOSUB, GOTO, IF, INPUT, LOAD, NEXT, ON, OUT, PLOT, POKE, PRINT, PUT, READ, REM, RESTORE, RETURN, SAVE, STEP, THEN, TO and WAIT. 3 command types: CONT, LIST and RUN. 19 mathematical functions: ABS(x), ATN(x), CALL(x), COS(x), EXP(x), FNx(y), FRE(x), INT(x), INP(x), LOG(x), PEEK(x), POS(x), RND(x), SGN(x), SIN(x), SPC(x), SQR(x), TAB(x), and TAN(x). 9 string functions: ASC(x$), CHR$(x), FRE(x$), LEFT$(x$I), LEN(x$), MID$(x$,I,J), RIGHT$(x$,I), STR$(x) and VAL(x$). 12 Disk File commands: COPY, DELETE, DEVICE, DIRECTORY, DAPLICATE, INITIALIZE, LOAD, READ, RENAME, RUN, SAVE, and WRITE.

CRT Terminal Commands: Page/Roll mode; Erase Line; Erase Page; Tab; Two Character Sizes; Blink; Cursor Home' Left, Right' up and Down; Cursor XY Addressing; Cap lock; CPU reset; Foreground/Background Color Selection; 15 Plot Modes; Blind Cursor Mode; Local, Full and Half Duplex Modes; Write Vertical Mode; and Transmit Cursor and Page Modes.

Available for the Compucolor II Programmed Sof-Disk Albums

Sampler (includes; 1. Demo Program of Sample Displays and CCII Features; 2. Game of Concentration; 3. One-Armed Bandit; 4. Biorhythms; 5. Loan and Repayment Schedule; 6. Memory Diagnostics for the CCII; 7. Engineering Application.)

Math Tutor: Math Tutor, Checkbook, Recipe Program, Math Dice, Biorhythms

Star Trek: Star Trek, Lunar Lander, Shoot, Tic-Tac-Toe

Hangman: Hangman, Math Tutor, Two to Ten

Chess: Chess, Acey Deucey, Line Five, Biorhythms

Othello: Othello, Math Dice, Concentration (Numbers), Concentration (Letters)

Text Editor

Assembler for the 8080
Some programs may require additional RAM memory.

Black Jack
Cubic Tic Tac Toe
Personal Finance Vol. 1 (Interest)
Personal Finance Vol. 2 (Loans)
Bonds and Securities
Equity
Personal Data Base
Income Tax-78

Maintenance Manual
Programmer's Manual

Mini Disk Drive
Uses 5¼" square Compucolor II diskettes.
Tracks: 40
Track Density: 48 tpi
Power on Delay: 1 second
Access Time: (average 20 tracks) 200 ms
Average Latency: 200 ms
Transfer rate: 76.8 Kilobits/sec.
Performance specifications:
Capacity formatted 51.2K Bytes/Side.
Both sides usable by flipping diskette over.

ROM Memory
Read Only Memory (ROM): 16K bytes of nondestructive read only. Memory sockets included for 8K bytes of additional EPROM/MROM memory. Includes DISK BASIC, File Control System, and Terminal Software.

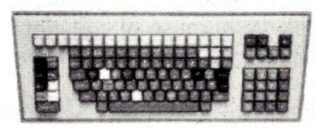
RAM Memory
Random Access Memory (RAM): 4K bytes for screen refresh. 8K bytes for user workspace. (Optional 16K and 32K — Models 4 and 5.)

BUS
50 pin bus: provides all addresses, data, clocks, etc., to allow the Compucolor II to be expanded with additional peripherals in the future.

Prices
Model 3 — 8K user RAM — $1,495 Retail*
Model 4 — 16K user RAM — $1,695 Retail*
Model 5 — 32K user RAM — $1,995 Retail*
Warranty 90 days

Available options
Second Compucolor II Mocro-Floppy™ disk drive — $400*
Expanded Keyboard: 101 keys with color and numeric clusters — $135*

Deluxe Keyboard: 117 keys, including 16 additional function keys — $200*



Additional 16K RAM Module (only for Model 3 and 4) — $375*

*U.S. Domestic Prices

T&W COMMUNICATION CENTERS

7231 FONDREN at SW FRWY. HOUSTON, TX 77036 (713) 774-9526

CIRCLE INQUIRY NO. 91

PROGRAM USAGE

During an actual emergency, the user selects item 1: EMERGENCY TREATMENT. This generates a master poison list with ten items classified into seven categories:

Caustics (Drano, lye, Liquid Plumber, etc.) 4
Drugs (aspirin, prescriptions, etc.) 6
Dyes (fabric, food, etc.) 2
Fuel (gasoline, kerosene, white gas, etc.) 2
Medicines (prescriptions, non-prescr, etc.) 6
Oils (heavy, thick lubricants, etc.) 3
Paint (thicker liquids, not thin aromatic) 2
Soaps (detergents, etc.) 5
Solvents (thinners, removers, etc.) 1
Ipecac 7

OR. . .type name of product or substance

At this point, the user enters either a category number from 1 to 7, or a product name. POISON responds first with the three emergency telephone numbers that were entered during the initialization step. Then, if a category number was entered, the emergency treatment is printed out. On the other hand, if a product name was entered, POISON first indicates that it is searching the disk library for the requested name. If the product name is located, the emergency treatment is printed out. If the name cannot be found in the library, the following message appears:

I cannot identify __ __ __ __
Try a related word, or a different spelling
of the same word.

The master poison list is reprinted, and the user can try again.

Experienced programmers will have little difficulty with POISON. There are, however, several places for the unexperienced to go wrong. The very first problem occurs when the user enters the number 1 and nothing happens. The solution is to remind the user to press RETURN after completing each entry. This simple step can be fatal in an emergency. Since POISON is written in BASIC, the problem can be easily remedied. Change lines 230, 280 and 420 of the source program to read:

```
230 INPUT"  Type a number (1-4), then RETURN",F
280 PRINT"MASTER POISON LIST, type the number and RETURN"
420 "  Type number or name, then RETURN",N$
```

A second serious problem can occur because the list of poisons is in uppercase letters. If the user enters a product name in lowercase letters, it won't be found. This problem can also be easily fixed by typing:

```
510 FOR I=1 TO LEN(N$)
512 N3=ASC(N$(I,I))
514 IF N3<97 THEN 518
516 N$(I,I)=CHR$(N3-32)
518 NEXT I
519 R1$=N$
```

The patch will convert each lowercase letter to the corresponding uppercase letter. Now the user can enter lye or LYE or Lye and POISON will find the entry.

THE USER'S MANUAL

The 24-page user's manual contains several parts. The sections needed during an emergency are clearly indicated. This insures that time will not be wasted by reading the disk-initialization and BASIC source-code sections when time is important.

The manual also has a helpful section on how to reduce the likelihood of poisoning, by keeping toxic materials and potentially dangerous materials such as aspirin, out of sight. It is also suggested that dangerous materials never be stored in empty food containers.

POISON looks like a useful addition to the library of good microcomputer programs. The applications are not limited to the home. It could be useful in business locations, especially those with unusual chemicals around.□

Generating Copies of TRS-80

The prerecorded machine-language tapes sold by Radio Shack for the TRS-80 present an interesting problem to the user: how to generate a backup copy of these tapes using the TRS-80. The general procedure for producing copies presented here uses the TRS-80 Editor/Assembler program as an example. The intent is to present to TRS-80 users a means by which they can protect their software investment; it is not to encourage piracy of Radio Shack's software.

A simplified flow chart is shown in Figure 1. The first two blocks, start address of source code and length of code, are of rather great importance and often difficult to determine. For example, T-BUG cannot be used to examine the Editor/Assembler for its start address since the programs reside in partially common memory space. However, it is reasonably well known that T-BUG resides in memory locations 4380H through 497FH. The Editor/Assembler was determined, through the courtesy of Radio Shack Computer Services, to reside in memory locations 4300H through 5D40H. The number of data blocks is computed next where the data block length can be from 1 to 256 bytes (256 bytes are designated as 00H). It is desirable to have the block lengths as large as possible to maximize the useful information transfer rate. Also, different data blocks can be the same length or different lengths.

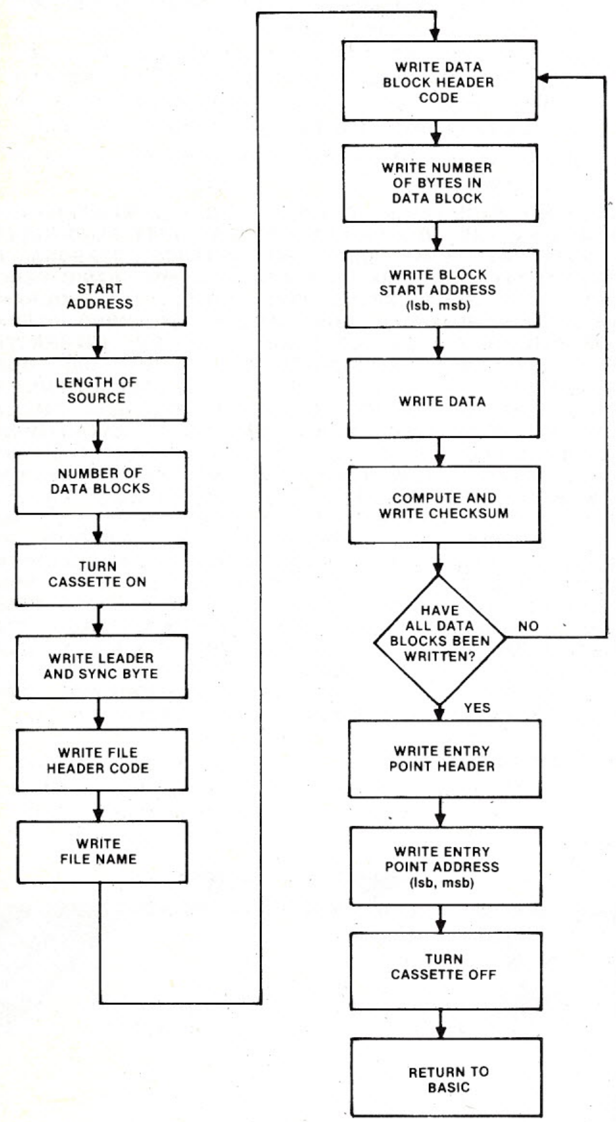


Figure 1. Flow Chart of Copying Program.

The tape recorder is turned on by either user generated code or more simply by using code in the LEVEL-II ROM. The calling code is as follows:

```
LD     A,00H; specify on-board cassette
CALL 0212H; define drive and turn on cassette.
```

The leader (255 bytes of zeros) and sync byte (A5H) can be written using the statement CALL 0287H or by equivalent code. The easiest method to output a byte to the cassette recorder is to execute a CALL 0264H with the byte to be recorded present in the A register. The file header code (55H) precedes the file name which comprises six bytes. If the file name, which must have a letter as the first character, is less than six characters, it must be padded with trailing blanks (20H).

Each data block comprises a data block header byte (3CH); a single byte specifying the number of data bytes in the data block; the starting address of the data block with the lsb first and msb second; the data bytes; and a one-byte checksum determined by summing the data bytes and the block starting address. The data blocks are output to the tape cassette recorder (Figure 1) until the data is exhausted.

The final information output is an entry point header (78H) and the entry point address (lsb, msb) which specifies the address that program execution begins. Executing CALL 01F8H will turn the tape cassette recorder off and JP 1A19H will re-enter LEVEL-II BASIC with the READY prompt. Alternatively, JP 0000H will produce the equivalent of the power-up condition.

Listing 1 presents a program, 137 bytes long, that will generate a backup copy of the TRS-80 Editor/Assembler. The comments provide adequate explanation of the program. Note, however, that there are other possible realizations of the program. For example, the file name could be defined using the DEFM statement and another write loop instead of the code from lines 00380 to 00490.

To produce a backup copy of the Editor/Assembler, load the Editor/Assembler and the program shown in Listing 1 using the SYSTEM command. Next, execute the copying program by entering "/28672" with the tape cassette recorder in the record mode and the volume setting between 5 and 6.□

Machine-Language Tapes

By R. B. Johnson

PROGRAM LISTING

```
            00100 ;THIS PROGRAM GENERATES A BACKUP
            00110 ;COPY OF THE TRS-80 EDITOR/ASSM.
            00120 ;THE EDTASM RESIDES IN LOCATIONS
            00130 ;4300H TO 5D40H.
            00140 ;THE PROGRAM BELOW HAS 7000H AS
            00150 ;ITS ORIGIN.
            00160 ;PROCEDURE: USING SYSTEM CMD, LOAD
            00170 ;EDTASM & THE PROGRAM BELOW.  THEN
            00180 ;ENTER "/28672" WITH TAPE RECORDER
            00190 ;SET FOR RECORD.
7000        00200       ORG     7000H
7000 210043 00210       LD      HL,4300H; START OF EDTASM IN MEMORY
            00220
7003 161C   00230       LD      D,1CH   ; SET UP FOR 28 LOOPS
7005 0E00   00240       LD      C,00H   ; CLEAR CHECKSUM REGISTER
0264        00250 W     EQU     0264H   ; ADDRESS FOR BYTE OUTPUT
            00260                       ; ROUTINE
7007 3E00   00270       LD      A,00H   ; SPECIFY ON-BOARD CASSETTE
7009 CD1202 00280       CALL    0212H   ; DEFINE DRIVE
700C CD8702 00290       CALL    0287H   ; WRITE LEADER AND SYNC BYTE
            00300                       ; LEADER IS 255-BYTES OF ZEROS
            00310                       ; SYNC BYTE CODE IS A5H
700F 3E55   00320       LD      A,55H   ; FILE NAME HEADER CODE BYTE
7011 CD6402 00330       CALL    W       ; OUTPUT BYTE TO TAPE RECORDER
            00340                       ; THE FILE NAME IS SIX-BYTES
            00350                       ; IN LENGTH.  FOR NAMES LESS
            00360                       ; THAN SIX BYTES, PAD WITH
            00370                       ; TRAILING BLANKS ( CODE 20H)
7014 3E45   00380       LD      A,45H   ; FILE NAME: EDTASM
7016 CD6402 00390       CALL    W       ; WRITE E
7019 3E44   00400       LD      A,44H
701B CD6402 00410       CALL    W       ; WRITE D
701E 3E54   00420       LD      A,54H
7020 CD6402 00430       CALL    W       ; WRITE T
7023 3E41   00440       LD      A,41H
7025 CD6402 00450       CALL    W       ; WRITE A
7028 3E53   00460       LD      A,53H
702A CD6402 00470       CALL    W       ; WRITE S
702D 3E4D   00480       LD      A,4DH
702F CD6402 00490       CALL    W       ; WRITE M
            00491 ;
            00492 ;
            00500                       ; THE LENGTH OF EDTASM CAN BE
            00510                       ; DIVIDED INTO AONE 1-BYTE DATA
            00520                       ; BLOCK AND TWENTY-EIGHT
            00530                       ; 240-BYTE DATA BLOCKS.  THE
            00540                       ; SINGLE BYTE BLOCK IS WRITTEN
            00550                       ; FIRST, FOLLOWED BY THE 240-
            00560                       ; BYTE DATA BLOCKS.
            00561 ;
            00562 ;
7032 3E3C   00570       LD      A,3CH
7034 CD6402 00580       CALL    W       ; WRITE DATA BLOCK HEADER CODE
7037 3E01   00590       LD      A,01H
7039 CD6402 00600       CALL    W       ; WRITE NUMBER OF BYTES IN THIS
            00610                       ; DATA BLOCK
703C 3E00   00620       LD      A,00H
703E CD6402 00630       CALL    W       ; WRITE LSB OF BLOCK START
            00640                       ; ADDRESS
7041 3E43   00650       LD      A,43H
7043 CD6402 00660       CALL    W       ; WRITE MSB OF BLOCK START
            00670                       ; ADDRESS
7046 7E     00680       LD      A,(HL)
7047 CD6402 00690       CALL    W       ; WRITE DATA
704A CD6402 00700       CALL    W       ; WRITE CHECKSUM
            00710                       ; THE CHECKSUM, REG. C, IS THE
            00720                       ; 1-BYTE SUM OF THE START ADDRESS
            00730                       ; AND DATA
704D 23     00740       INC     HL      ; HL IS POINTER TO THE EDTASM
            00750                       ; PROGRAM LOCATIONS.
704E 3E3C   00760 LP1   LD      A,3CH   ; GENERATE 28 DATA BLOCKS
7050 CD6402 00770       CALL    W       ; WRITE DATA BLOCK HEADER CODE
7053 3EF0   00780       LD      A,240
7055 CD6402 00790       CALL    W       ; WRITE, 240 BYTES IN THIS
            00791                       ; DATA BLOCK.
7058 7D     00800       LD      A,L
7059 CD6402 00810       CALL    W       ; WRITE LSB OF THIS DATA BLOCK
            00820                       ; START ADDRESS.
705C 7C     00830       LD      A,H
705D CD6402 00840       CALL    W       ; WRITE MSB OF THIS DATA BLOCK
            00850                       ; START ADDRESS.
7060 06F0   00860       LD      B,240   ; INITIALIZE INTER-LOOP COUNTER
7062 4D     00870       LD      C,L     ; INITIALIZE CHECKSUM WITH LSB
            00880                       ; OF START ADDRESS.
7063 7E     00890 LP2   LD      A,(HL)  ; LOOP TO OUTPUT DATA BYTES
7064 CD6402 00900       CALL    W       ; WRITE DATA BYTE TO TAPE
7067 81     00910       ADD     A,C     ; ADD CURRENT CHECKSUM TO DATA
7068 4F     00920       LD      C,A     ; STORE RESULT AS NEW CHECKSUM
7069 23     00930       INC     HL      ; BUMP HL UP BY ONE
706A 10F7   00940       DJNZ    LP2     ; GO TO LP2 IF NOT DONE
706C 79     00950       LD      A,C
706D CD6402 00960       CALL    W       ; WRITE CHECKSUM
7070 15     00970       DEC     D       ; DECREMENT BLOCK COUNTER BY ONE
7071 7A     00980       LD      A,D
7072 FE00   00990       CP      00H     ; IS BLOCK COUNTER ZERO?
7074 20D8   01000       JR      NZ,LP1  ; IF NOT ZERO, GO TO LP1
7076 3E78   01010       LD      A,78H
7078 CD6402 01020       CALL    W,      ; WRITE ENTRY POINT HEADER
            01030                       ; CODE (78H)
707B 3E8A   01040       LD      A,8AH
707D CD6402 01050       CALL    W       ; WRITE LSB OF ENTRY POINT
            01060                       ; ADDRESS.
7080 3E46   01070       LD      A,46H
7082 CD6402 01080       CALL    W       ; WRITE MSB OF ENTRY POINT
            01090                       ; ADDRESS
7085 CDF801 01100       CALL    01F8H   ; TURN CASSETTE OFF
7088 C3191A 01110       JP      1A19H   ; RE-ENTRY TO LEVEL-II BASIC
            01120                       ; IF YOU WANT TO RETURN TO
            01130                       ; THE EQUIVALENT OF A POWER-UP
            01140                       ; CONDITION, THEN CHANGE 1A19H
            01150                       ; TO 0000H
0000        01160       END
00000 TOTAL ERRORS
LP2     7063
LP1     704E
W       0264
```

A Format Approach to Structured Programming

Structural Decomposition

By Stanley Dunn
Department of Electrical Engineering
Drexel University
Philadelphia, PA 19104

INTRODUCTION

The computer field is in the midst of another revolution. Computer hardware is more abundant, and volumes upon volumes of software are, and will continue to be, flooding the market.

With the increase in the software industry, several astute individuals have realized that format organization and development philosophy is necessary to make programming an engineering discipline.

Since Dijkstra's original treatise on the subject, many authors have tried to present structured programming as either: programming without go-to's; top-down design; or modular programming. Structured programming is actually a combination of all of these rather than a singular concept.

This article introduces structural decomposition, which is a format approach to developing structured programs. There are three transformation rules associated with the process which will be presented and the entire process will be demonstrated by use of a small example.

THE STRUCTURAL DECOMPOSITION PROCESS

The basic idea behind this approach is that the designer considers the entire program as one large idea, and repeatedly breaks it down into smaller units followng the transformation rules. This is the part of the process known as *top-down design*. Initially, the program is considered to be one large block which is continually decomposed into smaller units, Figure 1.

The advantage to this approach is that immediately the beginning and the end of the program are known, with the steps in between being refined. This avoids groping through individual steps until an end is reached.

Modular programming design becomes important at this step. Each block interacts with the others only through the entry and exit points and no other way. Each module, or block, needs only to know what is available at the output of the previous module and what is to be presented at the input to the succeeding module, Figure 2.

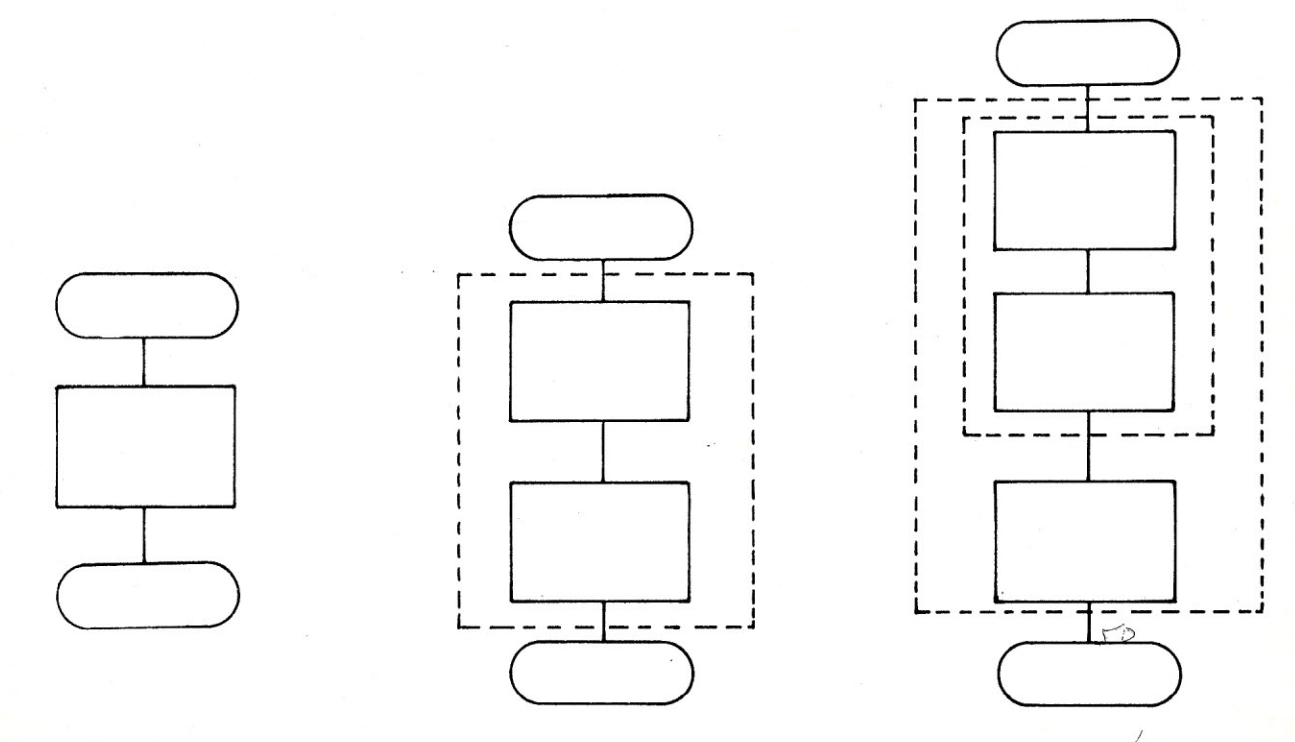


Figure 1. Transformation

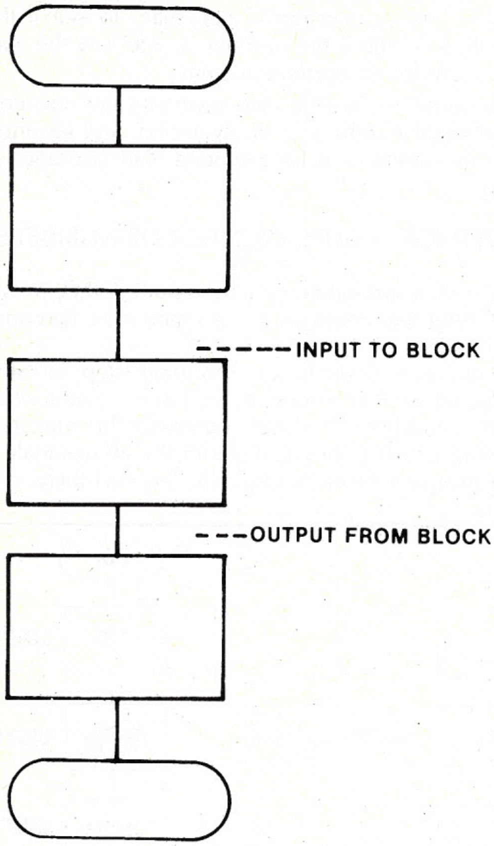


Figure 2. Sequence Block connections.

Therefore, each module can be designed and coded independent of the others; realizing that the module must present the data correctly to the succeeding module and take data correctly from the last module, and so forth.

This fact is extremely important in the structural decomposition process. Since each block is independent, the structure of the block can be changed, keeping the input and output the same without affecting the program. This consequently is the basic theory behind the three transformation rules of structural decomposition.

Before the transformation rules are presented, there is one final point to be covered and that is programming without go-to's. To do this effectively, the designer is limited to three constructs. These are: A sequential construct; a repetitive construct; a decision construct, Figure 3. Note that each has one entry and one exit point. This limits the jumping around in a program, and eliminates the need for a go-to or absolute jump construct.

The transformation rules of structural decomposition say that each sequence block, (rectangular block in flow chart) can be replaced by one of the three constructs mentioned above. **That is, one sequence block can be replaced by a series of sequence blocks, or one sequence block can be replaced by a decision block.** The final transformation rule is: **that a sequence block can be replaced by a repetitive construct,** Figure 4. The one-entry, one-exit property of each block is still maintained through the transformation.

In structural decomposition, the following steps are taken when writing the algorithm and subsequently the code. The program is first considered as one sequence block between the beginning and end. From the problem statement the designer determines how many blocks the program should be decomposed into. Each of these decomposed sections usually correspond to one of the following: Initialization, repetitive process, and printing of results.

After this initial decomposition, each sequence block is considered separately and is transformed using the transformation rules stated above. Traditionally, the most complicated is attacked first, so immediately the designer will be able to tell if the problem can be solved with the suggested procedure. If difficulties are found in transformation, the problem can be solved by trying a different algorithm.

This transformation process is continued until a level in the flowchart is reached which is suitable for coding the language being used. Obviously, for assembly language programming more transformation steps will be required than for programming in FORTRAN or BASIC.

A SMALL EXAMPLE

To illustrate the concepts introduced, consider the following problem:

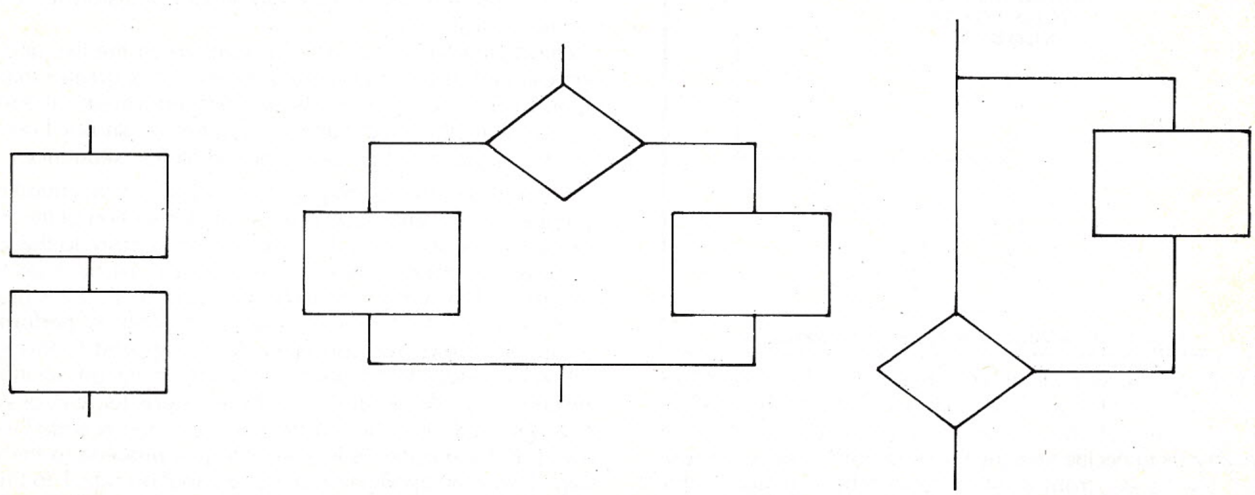


Figure 3. Legal constructs in programs.

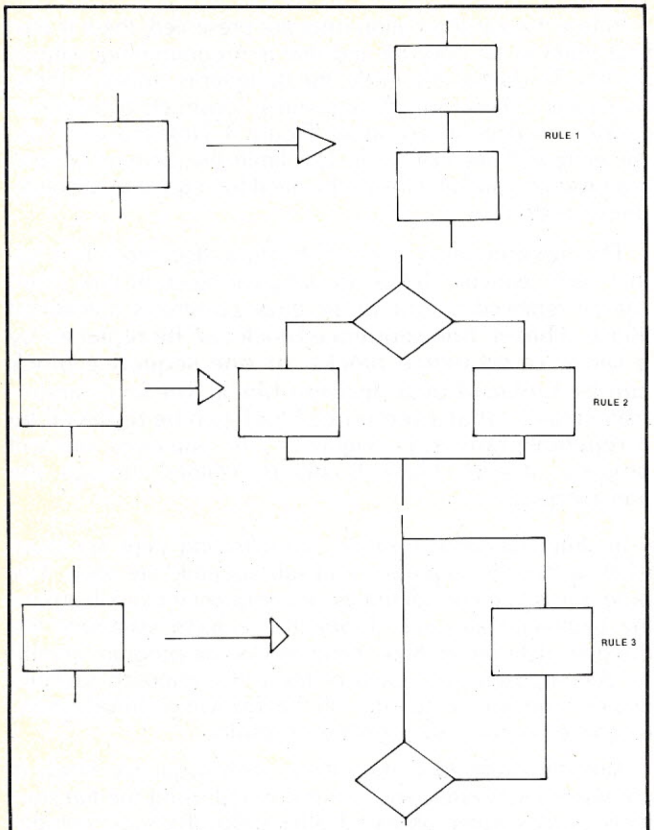


Figure 4. Transformation rules.

From a list of 25 numbers, compute the average of all the numbers that are greater than 5. The 25 numbers are already in memory, and they range in value between 0 and 9.

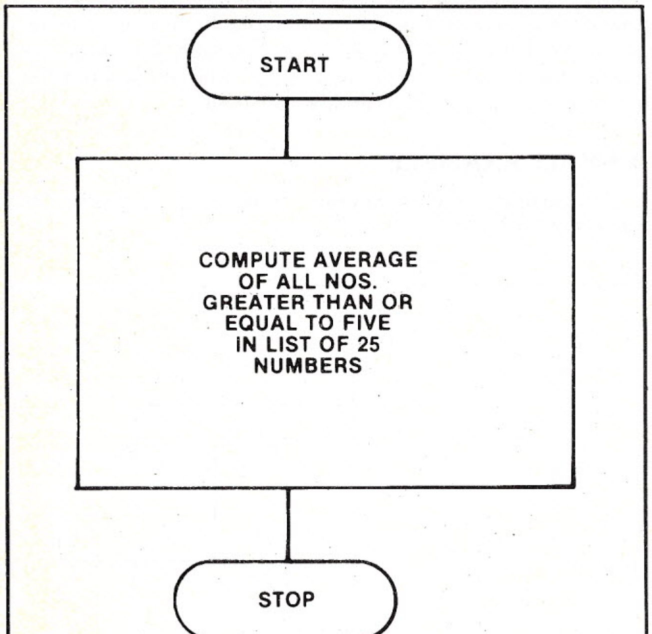


Figure 5. Initial program statement.

The above problem statement will serve as the initial block in the flow chart, Figure 5. At this point, the problem statement must be examined closer to pick out key words which will enable us to decide how the flow chart can be decomposed.

The first clause, from a list of 25 numbers, indicates that each one of the 25 numbers in the list will be checked. This means that a certain procedure will be repeated, and hence the need of a *repetitive construct.*

Of these 25 numbers, those that are greater than 5 will be averaged. First, a decision is necessary to see if they are greater than 5, and if the number is, add it to the sum, add one to count for computing average.

To compute an average, two quantities are needed. First, the number of numbers to be averaged; and secondly, the sum of the numbers to be averaged. The average is computed by:

AVERAGE = SUM/(NUMBER OF NUMBERS)

Since both a sum and count are required, they must be initialized. After the repetition, the average must be computed, then printed.

The above analysis is an important step in Structural Decomposition. It is necessary to have a point where the designer should *think ahead,* otherwise he may have no direction in which to proceed. From the above analysis, the original program block is decomposed into three sections, Figure 6.

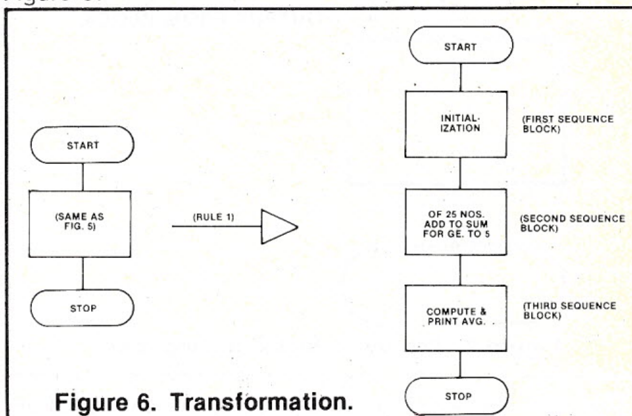


Figure 6. Transformation.

The think-ahead step showed that three stages were present in this problem. The first stage is the initialization of the variables needed. The second stage is the repetitive process and finally the output stage.

In general this will be the form of virtually any program encountered. This concludes the first stage of decomposition, now the transformation rules can be applied to further decompose the algorithm.

It was noted in the previous section that at this point the most complicated sequence should be handled. This, obviously, would be the middle sequence. Already from the initial analysis, this sequence block will be transformed into a repetitive structure.

The repetition is over the 25 numbers in the list, and the process performed each time is to see if it is greater than or equal to 5. If so, then a task will be performed; otherwise, just point to the next number. Figure 7a shows how the transformation rule has been applied to this sequence.

There is still transformation to be done on this middle sequence, but the emphasis is on a smaller section of the transformed sequence block. By referring once more to the initial analysis, the process block in the loop must be a decision construct. The number must be checked to see if it is greater than or equal to 5. If so, another process is performed. Figure 7b shows transformation rule 2 applied to this step.

It is necessary to go one step further in decomposition of the sequence. If the number in the list is greater than or equal to 5, the value is added to the sum and one is added to the count. If the number is less than 5, just proceed to the next step. This is an application of transformation rule 1 to the sequence block in the decision construct. Figure 7c shows in flow chart form the transformation rule applied twice, once on each side of the decision.

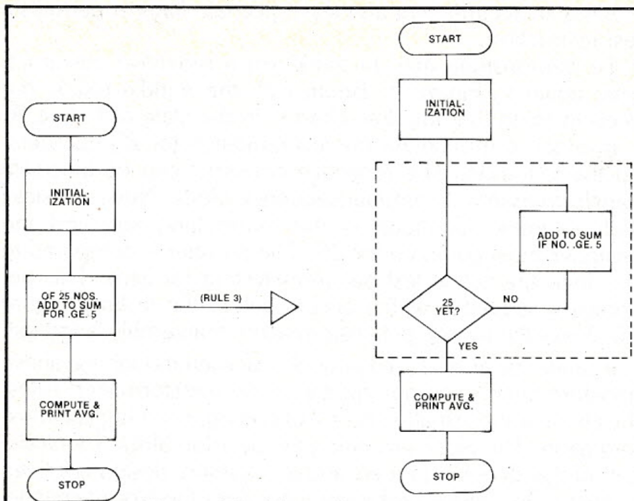


Figure 7a. Transformation Rule 3.

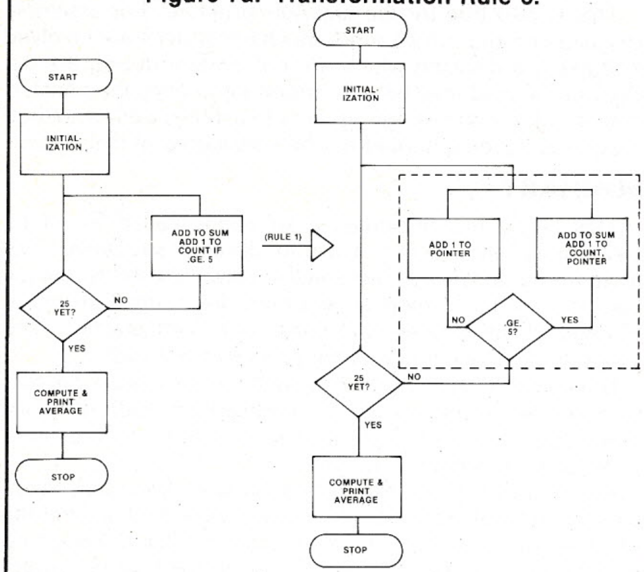


Figure 7b. Transformation Rule 2.

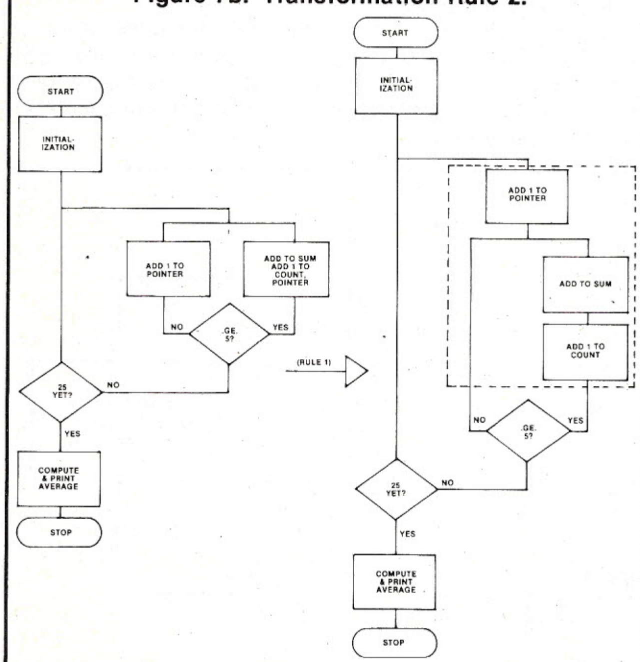


Figure 7c. Transformation Rule 1.

From the original decomposition, Figure 6, all that remains is to transform the first and third sequence. Since the first sequence is the initialization, each of the variables used should be set to the initial value necessary for running the program. The program begins pointing at the first number, so pointer = 1, also sum and count must be 0 since these two numbers will be used for running totals. Hence the first sequence block can be transformed by using the first rule. This is shown in detail in Figure 8a.

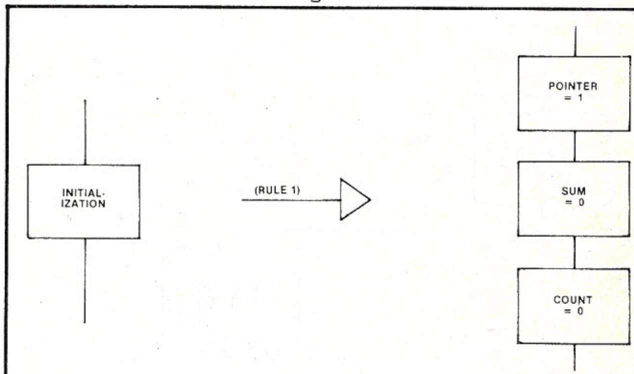


Figure 8a. Transformation of Initialization.

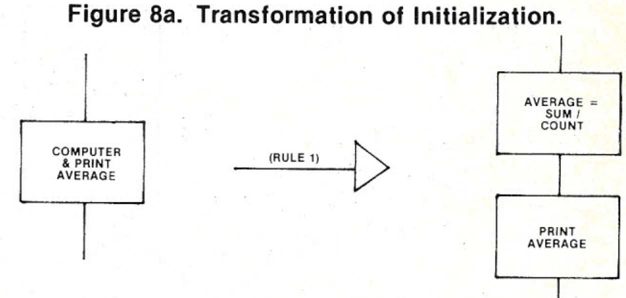


Figure 8b. Transformation of Third Block.

Now the third block of Figure 6 remains to be transformed. This is the final processing and output block. There are two parts to this block. First is to compute the average, then to print the average from the *think ahead* stage. Remember that the formula for the average is:

AVERAGE = SUM/COUNT

With this in mind the designer can use the first transformation rule to decompose the third sequence block. This is followed by the print statement. See Figure 8b for the decomposition.

Each of the transformed blocks can be put together now to yield the entire flow chart. At this point, the flow chart can be used to code the program in FORTRAN or BASIC, but will require further work to be coded in an assembly language. Figure 9 contains the complete flow chart to this point.

ADDITIONAL NOTES ON DECOMPOSITION TECHNIQUES

No doubt many will say that assembly language programs cannot be structured. The fault in this assertion lies in the fact that it refers to the code involved and not with the architecture of the program. Regardless of the language implemented, if the program exhibits good architecture, then it should be considered a structured program.

This is where go-to problems come into effect. It is nearly impossible to code a program in an assembly language without using a JMP (jump), or BRA (branch always). The go-to less stalwarts maintain that a program with a JMP or BRA is not structured, but again coding is not that important. It is the structure of the algorithm that is of prime importance.

There are several structures that are important to cover if they are to be implemented in an assembly language. These are the decision and repetitive construct since they both involve testing for certain conditions. This normally involves

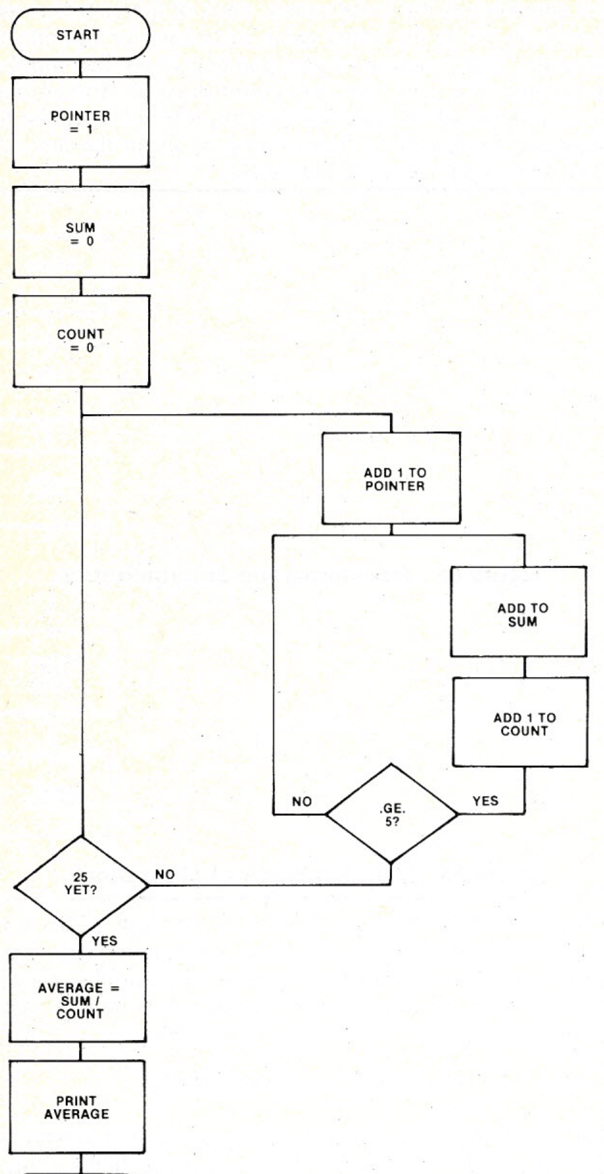


Figure 9. Final Flow Chart

loading an accumulator and testing certain flags or perhaps a test instruction.

To demonstrate how to transform a repetitive construct, refer again to Figure 6. Begin with the middle block, but keep in mind that the flow chart is in the state of Figure 9.

In order to implement the test condition for 25 numbers, do the following: The repetitive construct can be inversely transformed into its original sequence block. Now, consider that assembly language is the target language and the designer must compare for 25. The number is compared to 25, then the actual test becomes testing for zero. This becomes a sequence which occurs before the testing. Figure 10 shows the two stage transformation to assembly language.

It cannot be emphasized enough that even though the above transformation does not appear in the transformation rules, the above transformation is one of coding, not a logical transformation. The block preceding the decision block is logically part of that decision process and in algorithm design is treated as such. Only in the coding process does the block separate.

This is also true for the decision structure. For assembly language programming more mechanical steps are involved in making a decision which are not evident during the design, nor should they be! It is important to obey the transformation rules in the design process to achieve well structured programs. Coding should not be considered at this stage.

SUMMARY

This article has presented a technique called Structural Decomposition which is used to develop structured programs. An example was presented to show how this technique can be used to develop an algorithm for a small program. Mastery of this process will come only with practice. The programmer does not become proficient overnight.

But by applying this technique and the associated transformation rules, programs will be modularized, and also easily modifiable. This fact alone is reason enough to use these techniques in program design.

Assembly language transformation was discussed, keeping in mind that the logical structure is important and not the mechanical structure of the code involved. This is something that must not be lost sight of — this is probably where confusion arose in the past.

Structural Decomposition shows that structured programming is not just programming without go-to's, top-down programming, or modular programming, but it is the use of all three together which make structural programming a powerful development tool.□

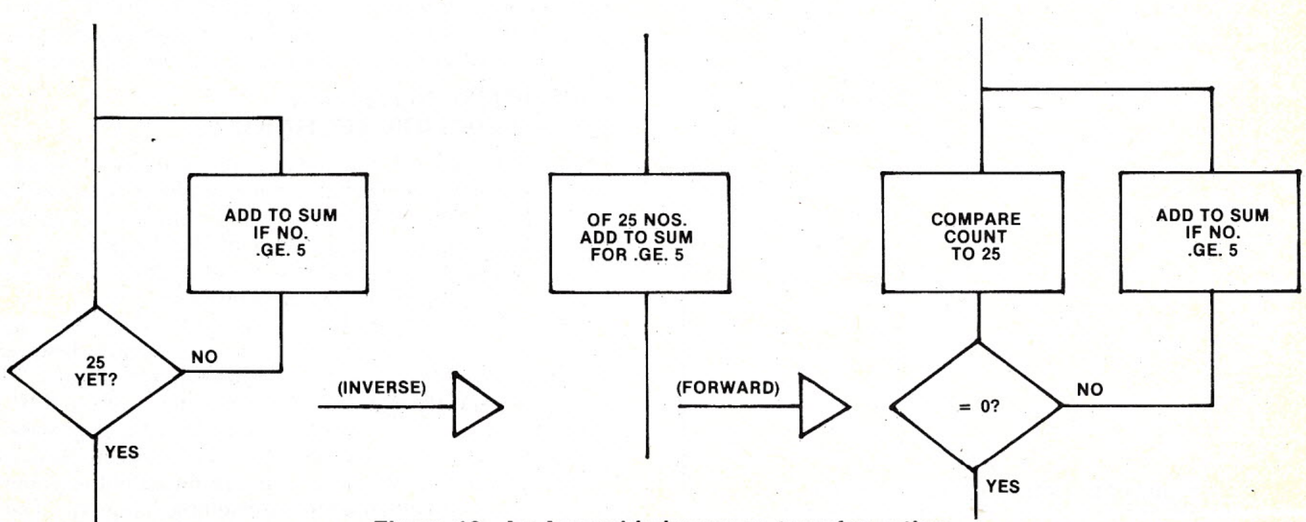


Figure 10. An Assembly language transformation.

AMERICAN TAX ASSOCIATES

Alpha Fedtax and Caltax

TAX TESTED

2 years in use

alpha micro systems

This is a thoroughly tested and successfully installed software package. It is very user oriented and simple to use. The package is as comprehensive as available computer tax services. It will calculate taxes, prepare and print all forms.

This package is supported by **American Tax Associates,** an established California accounting firm. In this way you can be assured that the yearly updates will be consistent with the current laws and accounting practices.

This package is a real time saver. It can perform income averaging automatically, and based on the data input, the program can determine whether to itemize or to use the standard deduction.

The client data collection and input procedures were selected based on the experiences of American Tax Associates, and the techniques used by many service companies. A simple form is completed during the client interview. The data from this form is later input into the computer for processing.

When the client data is entered into the computer you may select to have it print an audit trail of all data entered. This will enable you to double check the data entered.

The returns are printed on continuous preprinted IRS approved forms. Those forms not requiring a preprinted form are formulated and printed on blank paper. The data disk will hold up to 120 clients so the software is designed to print all of one page at a time.

The Alpha Micro system was chosen as the base computer system because of its multiuser capability, high throughput, and upward expandability into a hard disk system.

Yearly updates will be supported by American Tax Associates. These updates are available from either your dealer or directly from Mission Control.

SYSTEM REQUIREMENTS:

Language:	Alpha Micro Systems Basic (compiled)
Media:	8″ floppy diskette
CPU:	Alpha Micro AM-100
Memory:	64K RAM
Printer:	132 col with tractor feed
Floppy:	Dual 8″ drives required

Write for our brochure — Dealerships still available
Contact: Distributor

DISTRIBUTED BY **MISSION CONTROL**

2008 WILSHIRE BLVD., SANTA MONICA, CA 90403 • (213) 829-5137

ADVERTISER INDEX

*Manufacturer requests factory-direct inquiry.

MAGNETIC MEDIA SURVEY

INTERFACE AGE would like to solicit the help of readers to help us survey the market for magnetic media, tapes and disks. Please answer the following questions by circling the proper numbers on the reader service card.

CASSETTE TAPES

I have used the following brand(s):

212 Ampex
213 BASF
214 Memorex
215 Pryor
216 Radio Shack
217 TDK
218 Scotch (3M)
219 Other

I buy the following number of cassette tapes in an average month:

220 1-5
221 6-10
222 Over ten

I spend the following on cassette tapes per year:

223 5-25 dollars
224 25-50 dollars
225 50-100 dollars
226 Over 100 dollars

I use a cassette tape:

227 Once only
228 Several times

(This refers to the number of times a tape will be reused)

I generally buy the following length of tape:

229 15-minute (C-15)
230 30-minute (C-30)
231 60-minute (C-60)
232 90-minute (C-90)

My favorite brand of cassette tape is:

233 Ampex
234 BASF
235 Memorex
236 Pryor
237 Radio Shack
238 TDK
239 Scotch (3M)
240 Other

FLOPPY DISKS

I have used the following brand(s)

241 Ampex
242 BASF
243 Dysan
244 Memorex
245 Pryor
246 Scotch (3M)
247 Verbatim
248 Other

I buy the following number of diskettes in an average month:

249 1-5
250 6-10
251 Over ten

I spend the following on diskettes per year:

252 1-55 dollars
253 56-100 dollars
254 101-200 dollars
255 Over 200 dollars

I use a diskette:

256 Once only
257 Several times

(This refers to the number of times a diskette will be reused)

I generally buy the following type of diskette:

258 Hard sector
259 Soft sector
260 5.25 inch
261 8 inch

My favorite brand of diskette is:

262 Ampex
263 BASF
264 Dysan
265 Memorex
266 Pryor
267 Scotch (3M)
268 Verbatim
269 Other

PLEASE MARK YOUR ANSWERS

ON THE READER SERVICE CARD

Over the past two years customer acceptance of Vector Graphic microcomputers has just grown and grown and grown. It isn't terribly surprising, really. After all, it does make life a lot easier when you have reliable cost effective equipment.

So it is with our new Vector MZ business system with both data processing and word processing available all in one trouble-free system.

That's good because:

By having one interactive stand-alone system with optional software for all your business functions including Accounts Payable, Accounts Receivable, General Ledger, Financial Statement, Inventory and more, plus . . . word processing for mailing labels, text editing of manuals, catalogs, etc., collection letters, contracts and proposals – you'll have big business power.

Known throughout the industry for highly reliable equipment, Vector Graphic's new business system incorporates: the Z-80 based Vector MZ microcomputer with built-in dual-disk quad-density storage enough to handle as many as 4,000 customers and 12,000 inventory items; Vector Graphic's "Mindless terminal"; and a Centronics 120 character per second printer. Optional Diablo letter-quality printer is also available.

That's Vector Graphic – cost effective, versatile, reliable systems. When you get right down to it – nobody does it better.

COBOL NOW AVAILABLE

☐ Send me details immediately.

Name________________

Company________________

Address________________

City/State/Zip________________

Dealer inquiries invited.

VECTOR GRAPHIC INC.

Vector Graphic Inc., 31364 Via Colinas
Westlake Village, CA 91361, (213) 991-2302

See us at the NCC Show in New York, June 4, 5, 6 and 7, Booth No. 340, 341 and 342.

Small BUSINESS Wonder